RSRE
MEMORANDUM No. 4479

ROYAL SIGNALS & RADAR ESTABLISHMENT

A BI-DIRECTIONAL BAUDOT/ASCII CODE CONVERTER

Author: L Anderson

PROCUREMENT EXECUTIVE,
MINISTRY OF DEFENCE,
RSRE MALVERN,
WORCS.

CONDITIONS OF RELEASE

0105885 303454

************************* DRIC U

************************* DRIC Y

Reports quoted are not necessarily available to members of the public or to commercial organisations.

RSRE. MEMORANDUM No. 4479

A BI-DIRECTIONAL BAUDOT/ASCII CODE CONVERTER

L. ANDERSON

MAY 1991

SUMMARY

A unit has been developed to interface between a terminal generating BAUDOT Code at 75 BAUD and a terminal generating ASCII Code at 9600 baud, facilitating bi-directional serial communication. This report describes the hardware and software design; and includes the Operating Instructions.

INTENTIONALLY BLANK

CONTENTS

LIST OF FIGURES

APPENDIX

INTENTIONALLY BLANK

1. INTRODUCTION

In support of Operation Granby, IS5 Division of RSRE was required to produce a self-powered applique unit which would interface between a terminal generating Baudot characters at 75 baud (see Appendix A & C), and a terminal generating ASCII characters at 9600 baud (see Appendix B), providing bi-directional serial communication.

Due to the immediacy of the requirement, and the need for a flexible and low cost solution, an industry standard 8-bit Microcontroller was chosen to perform all data processing and interface control between the applique unit and the two terminals. This flexibility in design, resulted from the microcontroller's internal software control of all data handling functions and external interfacing protocols.

The unit was powered from a 240V AC supply, with two external 25-way D-type Cannon connectors: one to transmit and receive Baudot code characters and the other to transmit and receive ASCII code characters. A manual RESET button was also provided to initialise the unit on power up.

2. GENERAL

Details of the system configuration showing the interconnections between the unit and the two terminals are to be found in Figure 2.1. The unit contains two distinct processing paths. These are:-

1. The conversion from ASCII generated characters into Baudot characters.

2. The conversion from Baudot generated characters into ASCII characters.

Due to the different baud rates of the two terminals, it was necessary to implement a scheme to provide simultaneous transmit and receive capabilities. This was achieved by routing the data transmit line from each terminal to a different microcontroller for conversion into the opposite serial format, each microcontroller operating asynchronously and independently of the other.

It can be seen from Figure 2.1 that microcontroller 1 converts from ASCII into Baudot and microcontroller 2 converts from Baudot into ASCII, indicating that each microcontroller transmits a different type of serial data stream from that which it receives, and at a different baud rate. Each microcontroller was configured to be interrupt driven on receipt of an ASCII or Baudot character. The ASCII characters were received and transmitted via a dedicated software configurable serial port, resident on chip, while the

Baudot characters were received and transmitted via a microcontroller port pin, configured in software to act as a pseudo serial port; where the received characters were identified using waveform sampling routines and the transmitted characters were produced using Pulse Width Modulation (PWM) routines.

3. HARDWARE DESCRIPTION

Figures 2.1 & 3.1 show the system hardware employed. The hardware operates from a single 5 Volt supply, derived from an 240V to 5V AC-DC converter. From Figure 3.1 it is evident that the converter contained two basic functional elements:-

1. The RS232 serial interface (MAX232)

These MAX232 devices are RS232 communications interfaces, used to generate the positive and negative logic levels required for true RS232 communication from a single 5V supply. Therefore, with the aid of an on chip charge pump, the MAX232 generates -10V to represent a Logic 1 and +10V to represent a Logic 0 (see Appendix B).

2. 80C51 INTEL MICROCONTROLLER

This is a 8-bit processing device which contains, 8K bytes of EPROM and 256 bytes of user addressable RAM. It also has four 8-bit bi-directional ports, Port 3 having a dedicated software programmable serial interface and a number of external interrupt pins, where each of the 32 individual port pins are uniquely addressable.

To generate the two different baud rates required for this application, namely 75 and 9600 baud, an 11 592 MHz crystal was used, to allow the internal timer which generated the appropriate data bit timings to be loaded with a convenient divide ratio.

To receive and transmit characters at different baud rates would have required each microcontoller to contain two dedicated software configurable serial interfaces. However each microcontroller had only one dedicated software configurable serial port. Therefore to solve this problem, and provide a flexible and re-configurable solution, a second serial interface was simulated, using an external interrupt pin (INT1), which would under software control:-

1. On Receive:-
   Be interrupted by the START bit of the Baudot character serial stream, sample the serial stream and extract the 5 data bits (see Appendix A) at 75 baud.

2. On Transmit:-
   Generate a Baudot character serial stream using software controlled Pulse Width Modulation (PWM) at 75 baud.

Microcontroller 1's function of receiving ASCII characters at 9600 baud, converting to Baudot and re-transmitting to the Baudot terminal configured to 75 baud caused a data bottle neck. If the length of message expected could have been defined and would never exceed the available Ram space, a FIFO buffer could have been used to store the characters awaiting transmission, until the processor was free. Since there was no constraint on the message length this was not possible. So the approach adopted was to use the Clear To Send (CTS) line from the ASCII terminal in conjunction with the large character buffer built into the terminal. The CTS line was controlled to hold-off receipt of more characters until the processor had carried out the conversion and re-transmitted the

character at 75 baud. Control of the CTS line was exercised via a buffered port pin with an additional, external, transistor used to provide sufficient current drive capability.

4. SOFTWARE DESCRIPTION

The software controlling the converter function resides in three programs: one program resident in microcontroller 1, which converts from ASCII into Baudot; one program resident in microcontroller 2, which converts from Baudot into ASCII; and the third program which defines all user constants and Eprom look-up tables, which resides in both microcontrollers. This system Firmware resides in the 8K bytes of EPROM within the microcontrollers.

4.1 Microcontroller 1 : ASCII to BAUDOT Conversion

The program listings are contained in Appendix E & G, with the Structured Design Methodology (SDM) charts contained in Figures 4.1.1 to 4.1.13.

4.1.1 Main Program Routine (ASCII_BAUDOT_CONVERT)

Details of the main program routine are shown in Figure 4.1.1. On RESET the processor jumps to the starting address of the main routine via the reset vector (located at address 0000H in the on-chip Eprom). Program execution begins by configuring the microcontroller's ports to the desired input/output conditions (see Figure 4.1.2). The serial port is then configured to allow serial communication to and from the ASCII terminal (see Appendix A).

As the program is interrupt driven, the processor only responding if a serial interrupt occurs, the serial interrupt flag must first be activated and enabled.

The main program routine then enters an infinite loop where its only action will be to vector to the appropriate serial interrupt service routine (ISR) on receipt of an ASCII character.

4.1.2 Serial Interrupt Service Routine (SER_ISR)

When a serial interrupt occurs the processor vectors to the serial interrupt service routine and tests for character validity (see Figure 4.1.3). On receipt of a valid ASCII character the processor inhibits, by the assertion of the CTS line, the ASCII terminal from transmitting any further characters from its buffer. Conversion from ASCII into Baudot then occurs, using the ASCII value of the received character to address its Baudot equivalent in Eprom (see Figure 4.1.4).

Appendix A shows that the Baudot characters are divided into either upper or lower case, to achieve more than 32 (2^5) possible characters. To switch between these cases a special character, either a Lower Case Shift character (L.C.S.) or an Upper Case Shift character (U.C.S.), has to be issued prior to the transmission of a character. Failure to transmit the appropriate special case character will result in the Baudot terminal decoding the received character incorrectly. As an example consider the character stream "ABC1A", where nothing has been sent prior to this. The transmission would then be of the form:-

L.C.S.	A	B	C	U.C.S.	1	L.C.S.	A

Therefore the processor determines whether a case shift character is necessary prior to the character (see Figure

4.1.4 to 4.1.6) and outputs the serial stream to the destination Baudot terminal.

Finally the processor clears the CTS line, allowing the ASCII terminal to send another character from its buffer.

4.1.3 Baudot Serial Transmission

As each microcontroller contained only one dedicated serial interface, but was required to transmit characters at a different baud rate from those it received, it was necessary to simulate a second using a port pin and software controlled PWM routines (see Figure 4.1.7). The modulation scheme used Timer 1 (see Figure 4.1.8 & 4.1.9) to generate a strobe at half data bit intervals (6.65mS at 75 Baud), defining sequential bit intervals which marked the transition between:-

1. Start bit and Data Bit D0.
2. Data Bits D(x) and D(x+1) where x = 0..3.
3. Data Bit D4 and Stop Bits.

As each bit forming the Baudot serial stream is sent, (see Figures 4.1.10 to 4.1.13) the appropriate logic levels are output depending whether a Start, Stop or Data Bit is required.

4.2 Microcontroller 2 BAUDOT to ASCII conversion

The program listings are contained in Appendix F & G, with the Structured Design Methodology (SDM) charts contained in Figures 4.2.1 to 4.2.10.

4.2.1 Main Program Routine (BAUDOT_ASCII_CONVERT)

Details of the main program routine are shown in Figure 4.2.1. On RESET the processor jumps to the starting address of the main routine via the reset vector (located at address 0000H in the on-chip Eprom). Program execution begins by configuring the microcontroller's ports to the desired input/output conditions (see Figure 4.2.2). The serial port is then configured (see Appendix B) to allow serial communication to and from the ASCII terminal once a character has been received from the Baudot terminal and converted.

Although this program, like the previous, is interrupt driven, the interrupt which flags the processor indicating that a Baudot character has been received is the external interrupt pin (INT1) and not the serial interface. This pin is configured in software, to sample the incoming serial stream from the Baudot terminal, and interrupt when the Start bit of each serial character packet is detected (see Appendix B). As part of the main program routine the external interrupt 1 is both enabled and activated.

The main program routine then enters an infinite loop where its first action will be to vector to the external interrupt 1 service routine when the detection of the Start bit causes the processor to be interrupted.

4.2.2 External Interrupt 1 Service Routine (INT1_ISR)

When the Start bit, which marks the beginning of a serial character packet from the Baudot terminal, causes an interrupt, the processor vectors to the interrupt 1's ISR (see Figure 4.2.3). The processor then samples the serial stream in the middle of each bit frame and extracts the corresponding 5 data bits which constitute a Baudot character (see Figures 4.2.4 to 4.2.6).

As an example, take the receipt of Baudot character "A" which has bit pattern 00011. The corresponding sample waveform would be as found in Appendix D. From this diagram it is clear that the processor first aligns itself to the middle of the Start bit, labelled S0, then takes 5 successive samples separated by the incoming serial bit rate (13.3mS) to extract the Baudot character "A".

When the Baudot terminal transmits characters it also transmits, depending on whether upper or lower case is required (see Appendix A), one of two case shift characters. If any of these characters are retrieved the processor waits for the next serial packet, which is the actual character, and retrieves the corresponding 5 data bits. It then uses these data bits and the case shift character to address Eprom and extract the ASCII equivalent character. If no case shift character is received the processor converts the current 5 data bits into the corresponding ASCII equivalent character without waiting for the next serial packet. This decision process can be seen in Figure 4.2.3.

4.2.3 Transmission of ASCII Character (PRT_CHR)

In this system where microcontroller 2 receives at 75 baud and transmits at 9600 baud, no data buffer is required. This is because the converted character would be transmitted onto the ASCII terminal before (even if the worst case of 2 Baudot characters, each being separated from the other by the bit rate (13.3mS) is anticipated) the next character from the Baudot terminal, could possibly be received.

Therefore having converted the received Baudot character into its ASCII equivalent, the character is loaded into the serial buffer (SBUF) and transmitted (Figure 4.2.8).

5. CONCLUSIONS

In rapid response to an Operation GRANBY requirement, IS5 Division has produced a compact, low cost, reliable applique unit to allow direct connection between an ASCII terminal and a Baudot configured terminal.

This equipment can be easily reconfigured to operate at any baud rate by minor software alterations without the need for any hardware re-design.

6. ACKNOWLEDGEMENTS

The author would like to thank all members of the E.W. System Techniques section for their assistance and advice during the design and assembly of the applique unit.

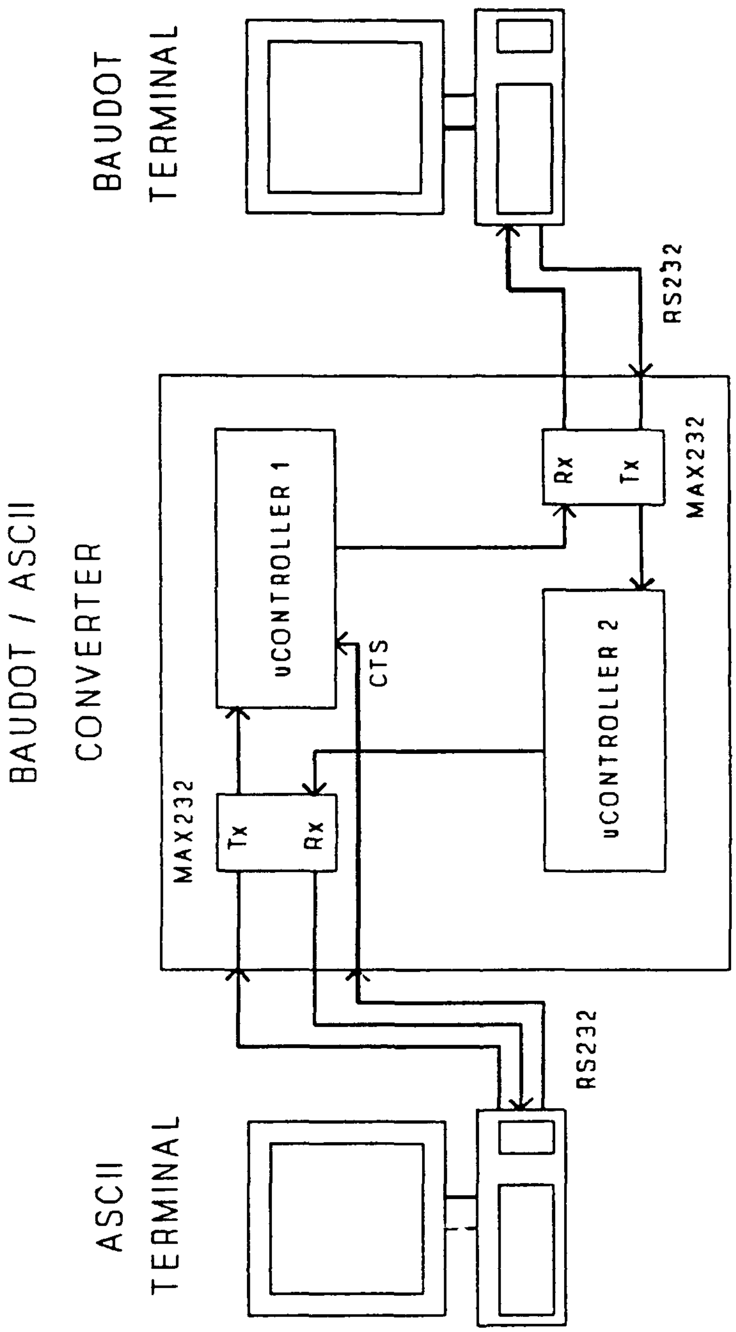


FIG 2.1 SYSTEM DIAGRAM

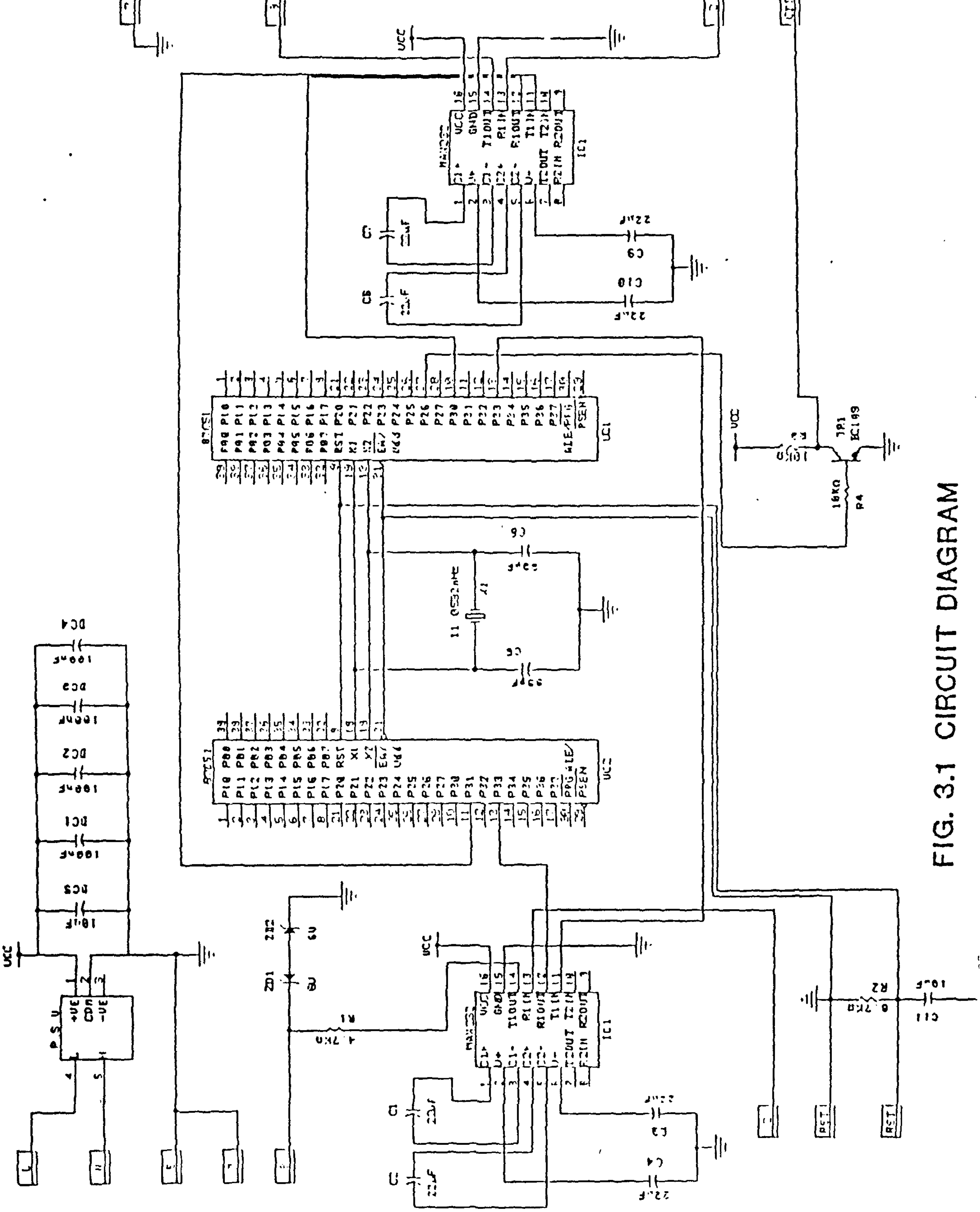


FIG. 3.1 CIRCUIT DIAGRAM

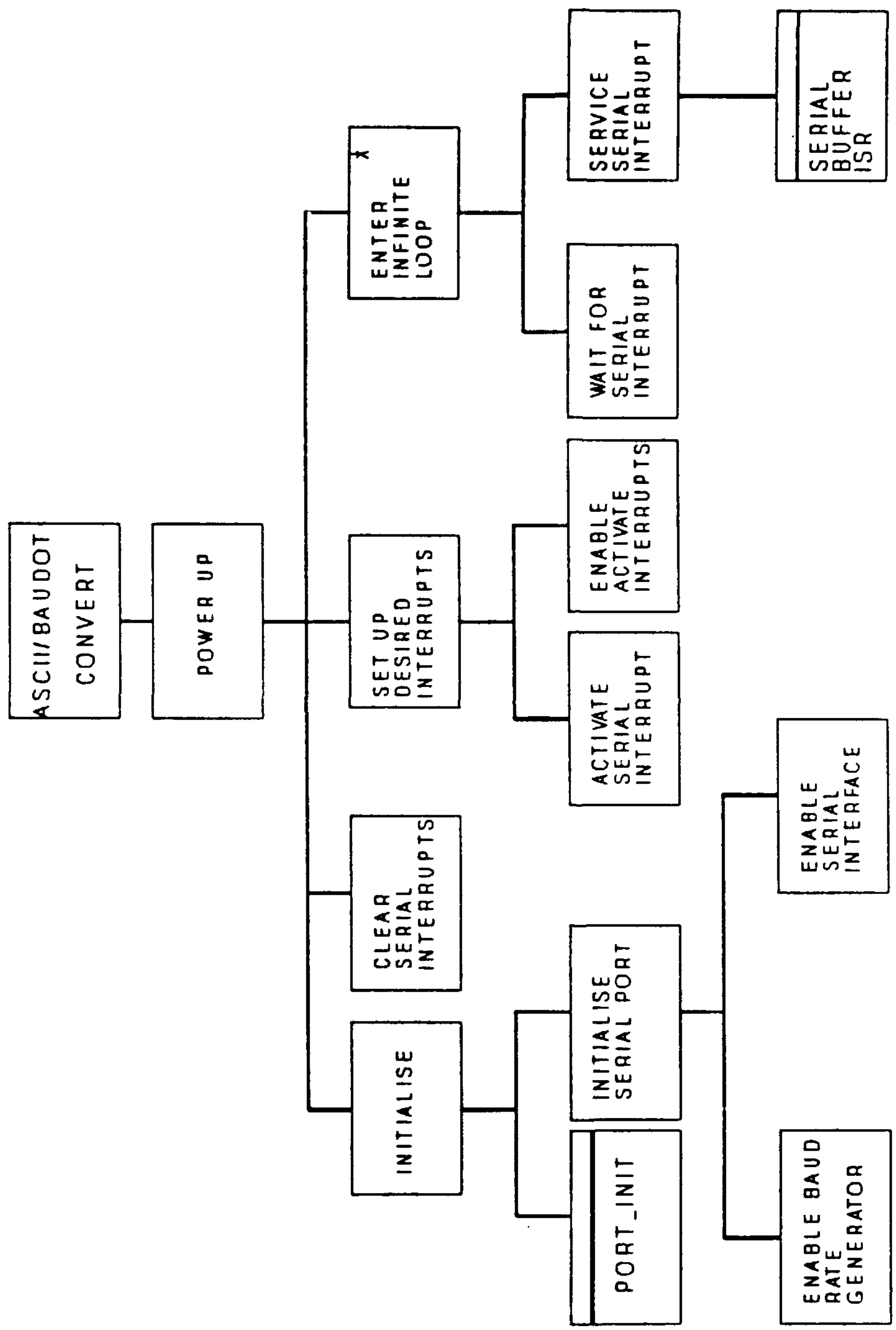


FIG 4.1.1 MICROCONTROLLER 1 MAIN ROUTINE

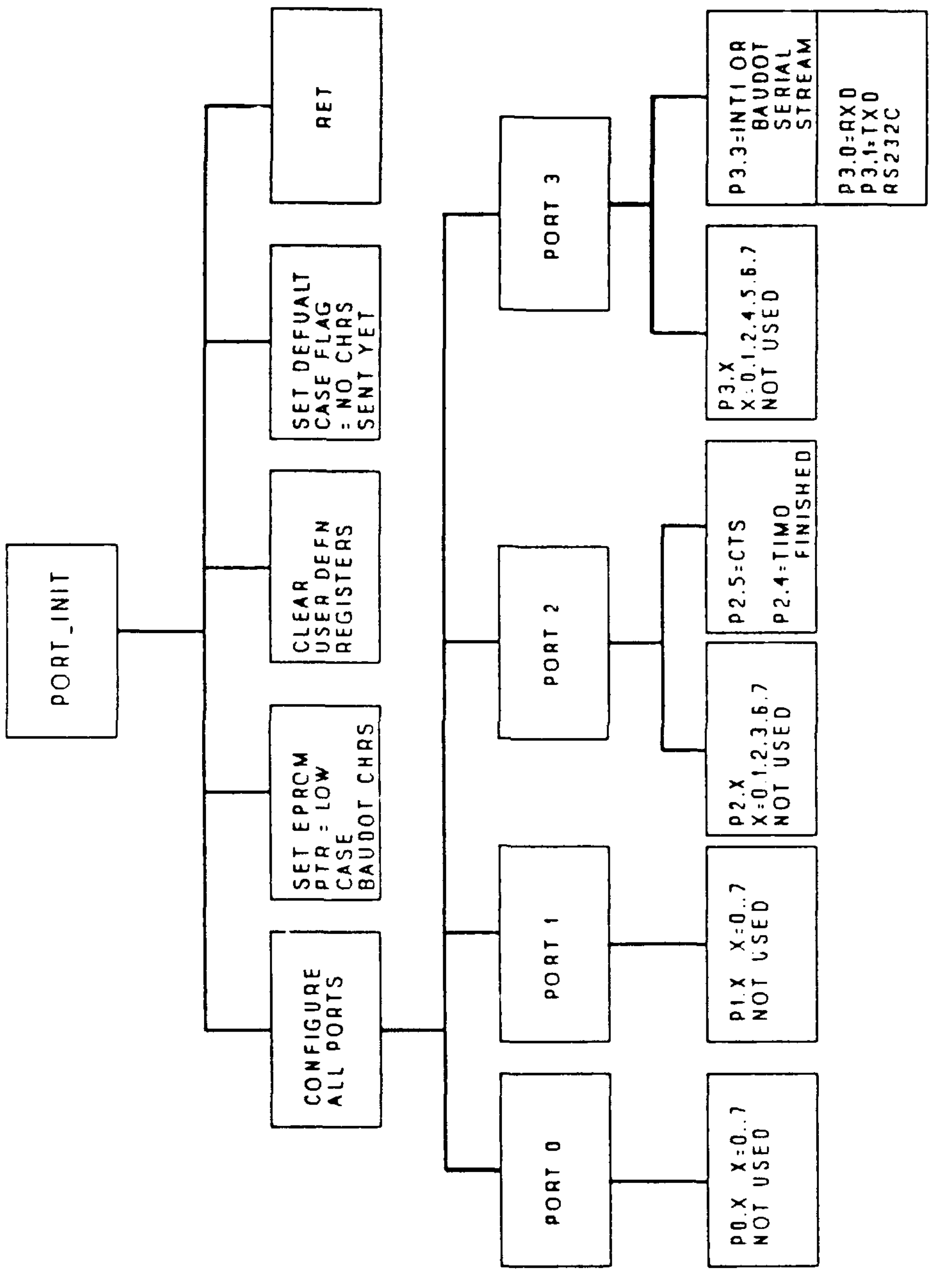


FIG 4.1.2 ROUTINE TO INITIALISE THE MICROCONTROLLERS PORTS

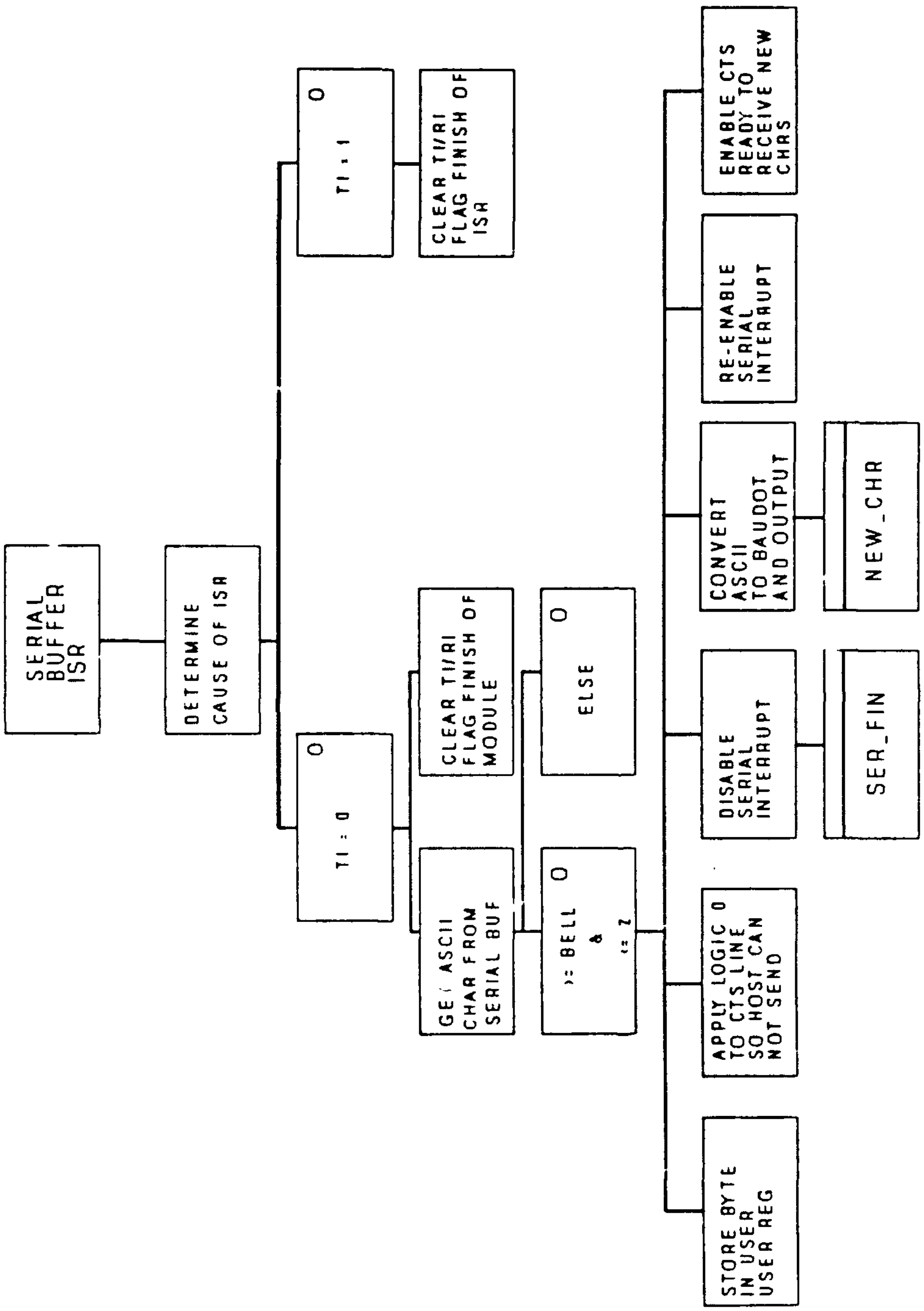


FIG 4.1.3 SERIAL INTERFACE INTERRUPT SERVICE ROUTINE

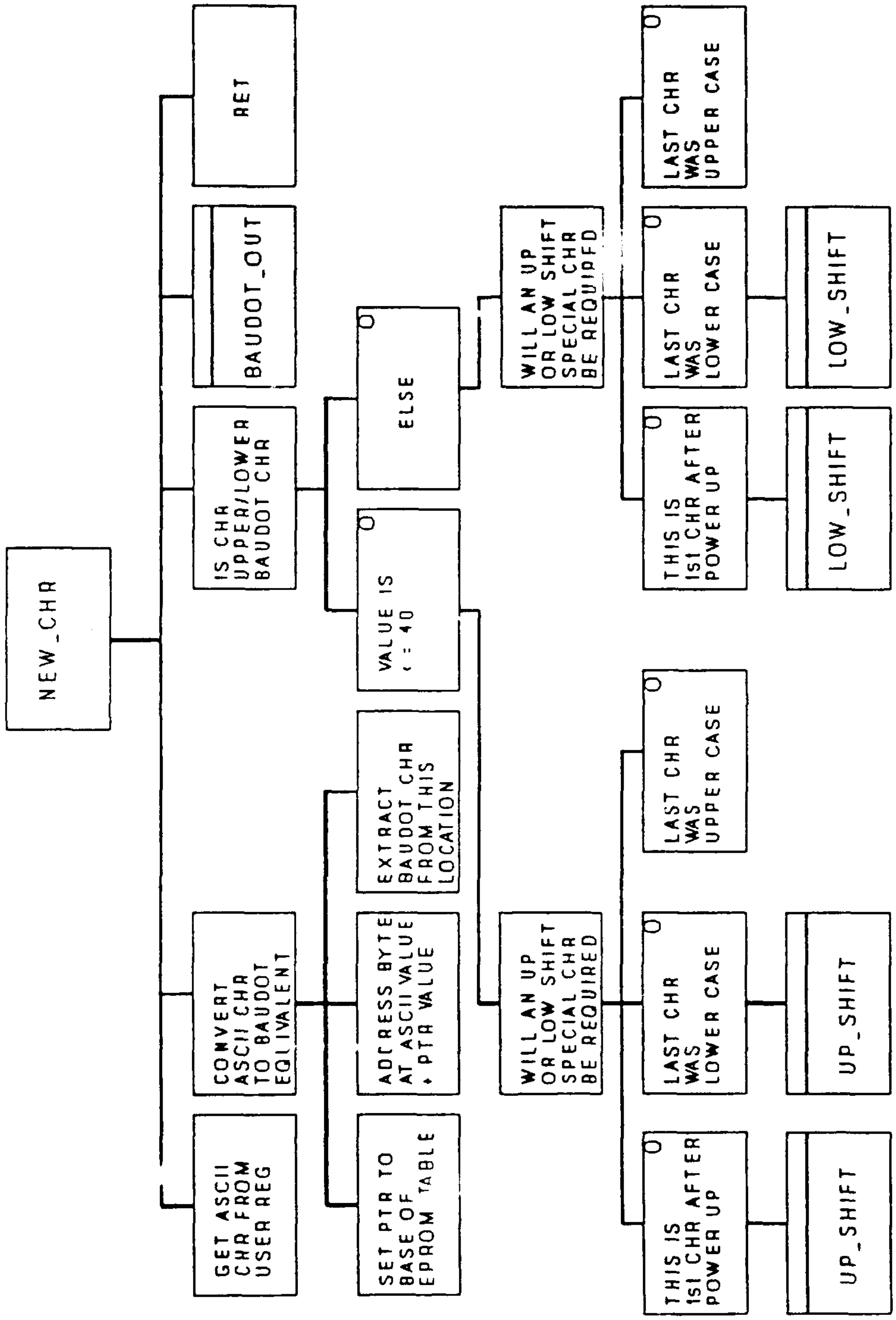


FIG 4.1.4 ROUTINE TO CONVERT FROM ASCII INTO BAUDOT AND TX

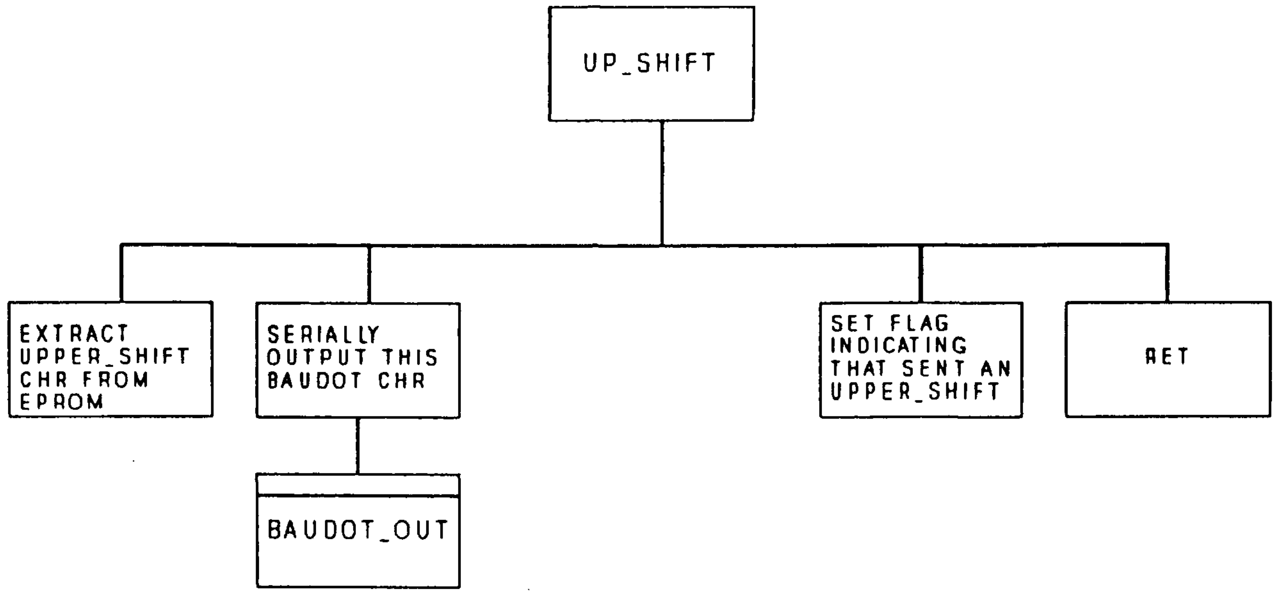


FIG 4.1.5 ROUTINE TO O/P AN UPPER CASE SHIFT CHARACTER

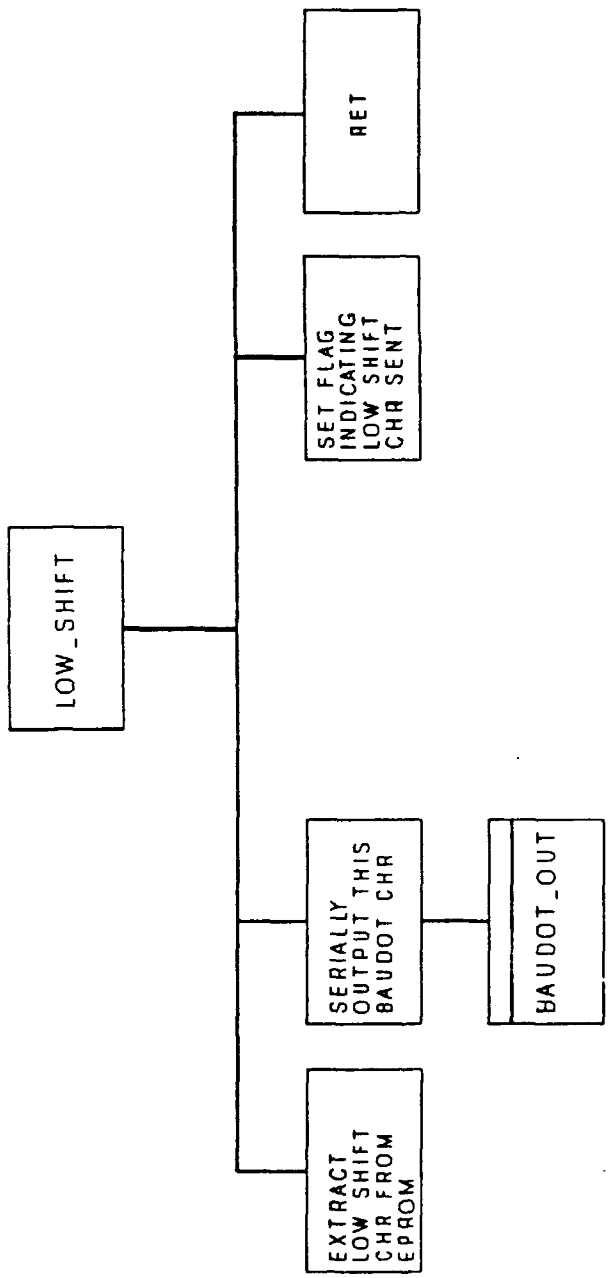


FIG 4.1.6 ROUTINE TO O/P A LOWER CASE SHIFT CHARACTER

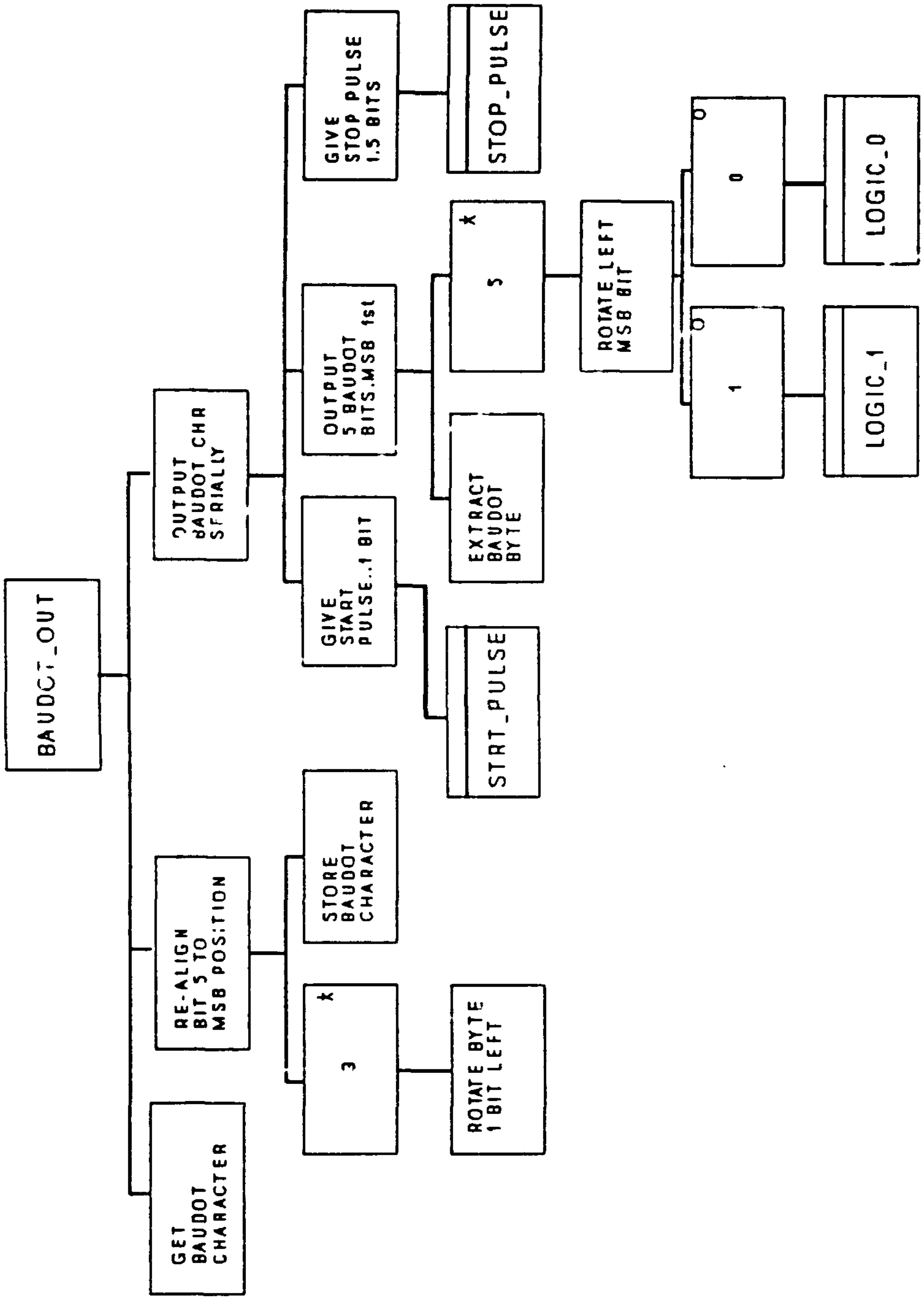


FIG 4.1.7 ROUTINE TO SERIALLY O/P A BAUDOT CHARACTER

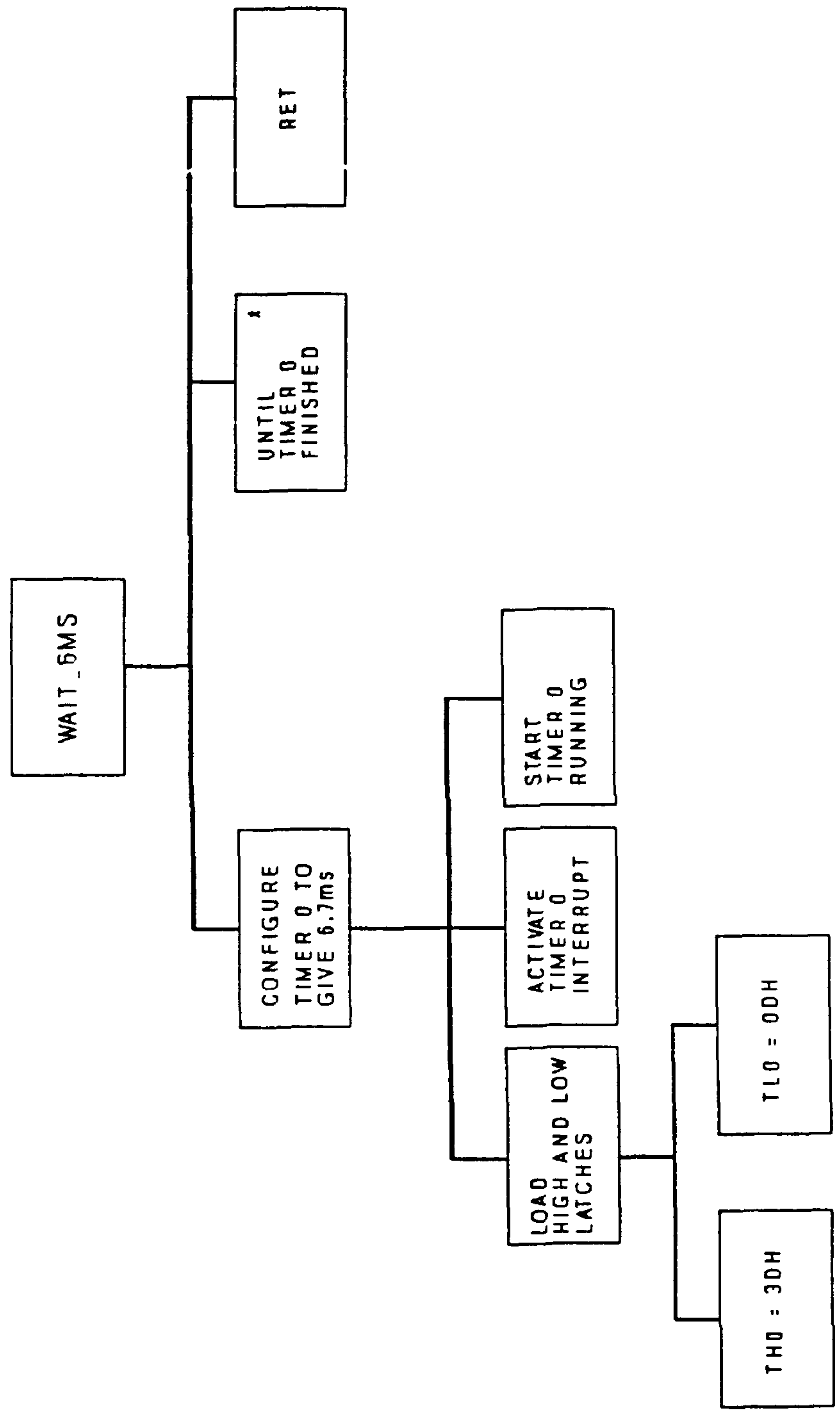


FIG 4.1.8 ROUTINE TO GENERATE A 6.65mS TIMING INTERVAL

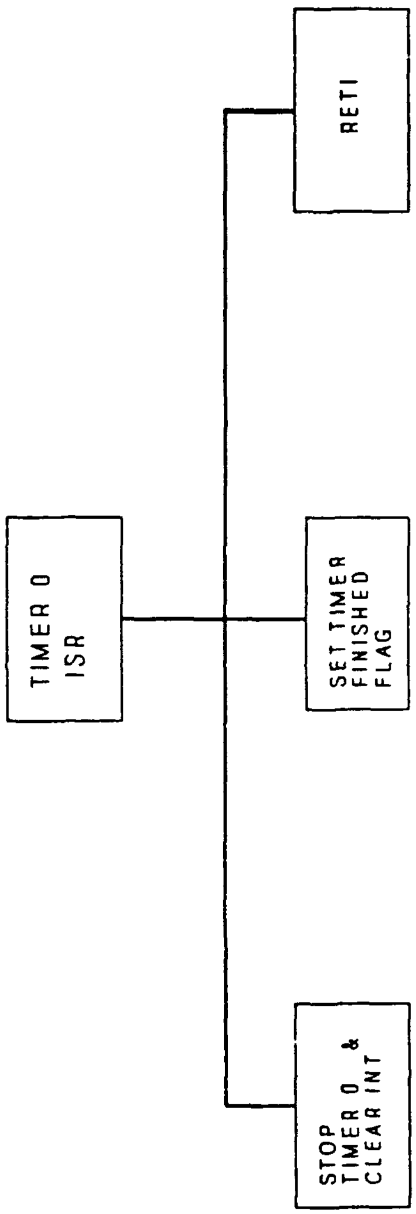


FIG 4.1.9 TIMER 0 INTERRUPT SERVICE ROUTINE

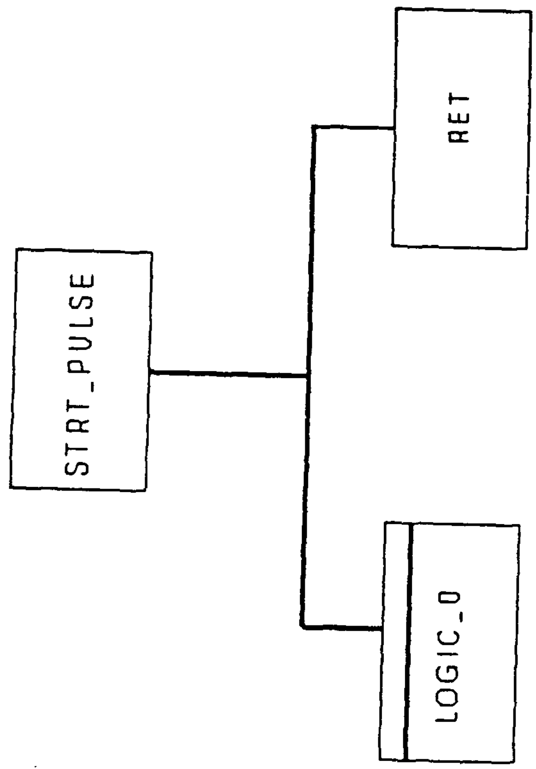


FIG 4.1.10 ROUTINE TO GENERATE A START PULSE

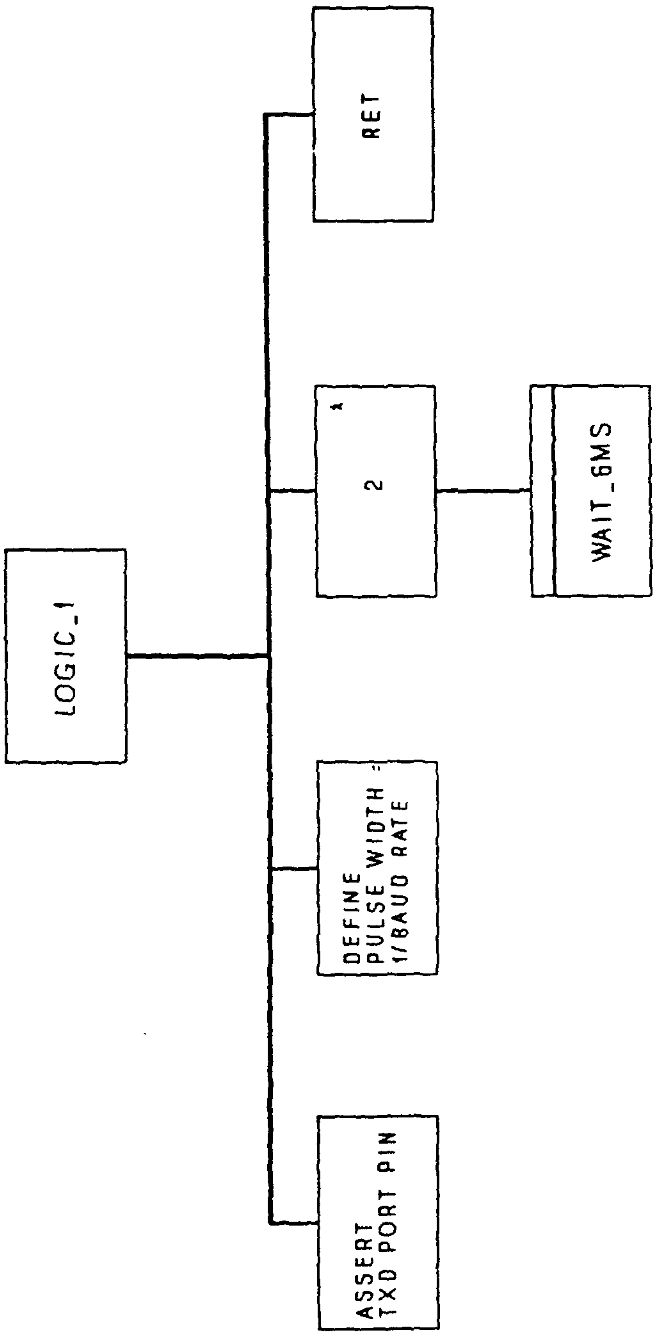


FIG 4.1.11 ROUTINE TO GENERATE A LOGIC 1

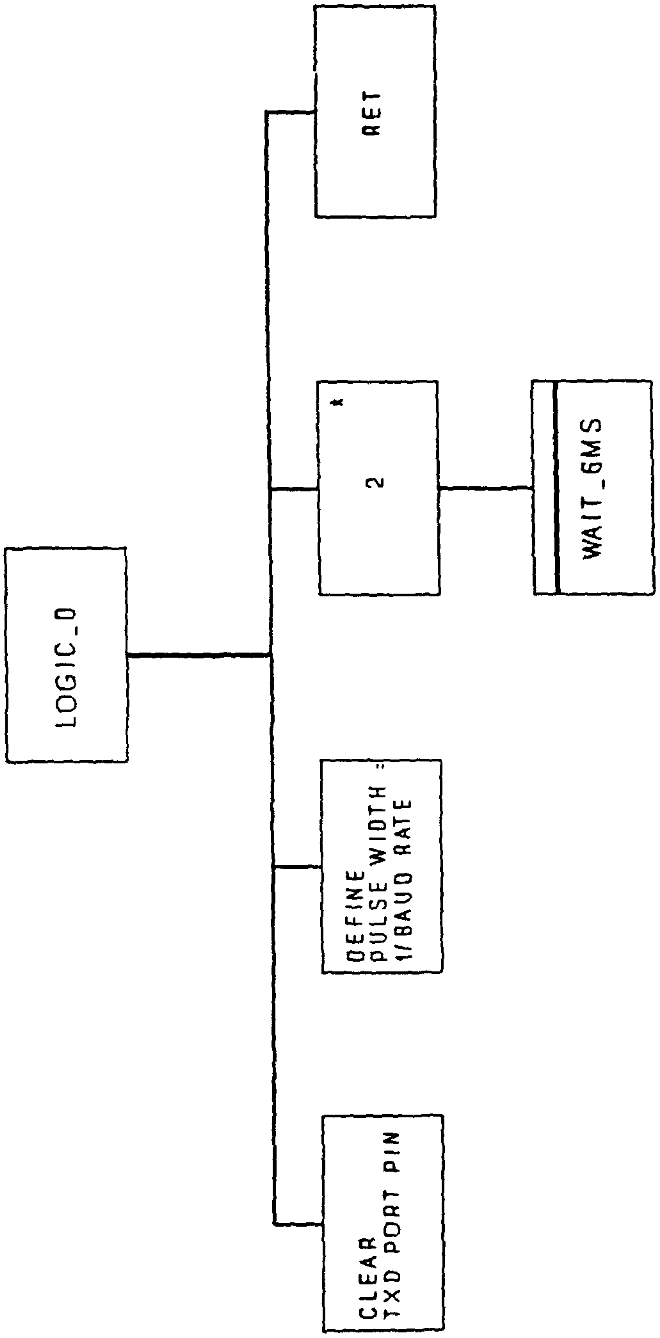


FIG 4.1.12 ROUTINE TO GENERATE A LOGIC 0

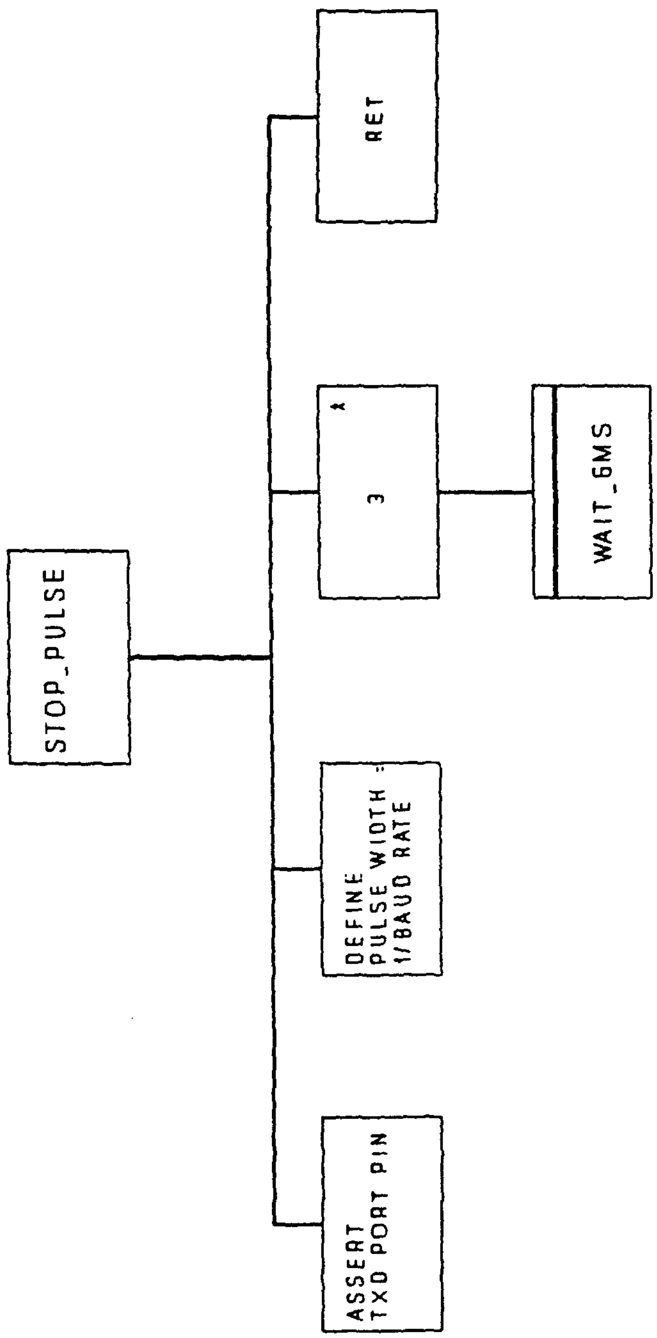


FIG 4.1.13 ROUTINE TO GENERATE A BAUDOT STOP PULSE

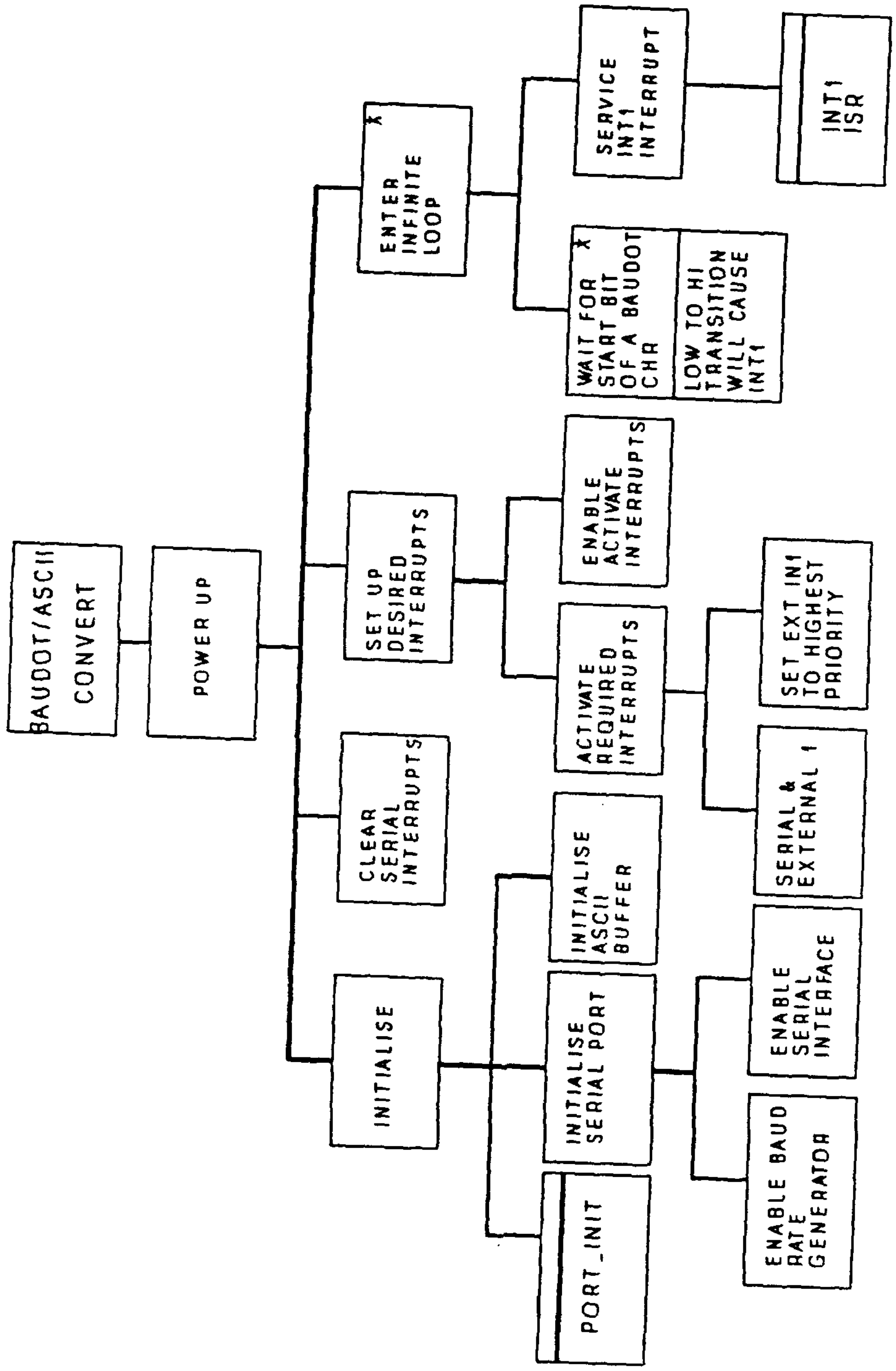


FIG 4.2.1 MICROCONTROLLER 2 MAIN ROUTINE

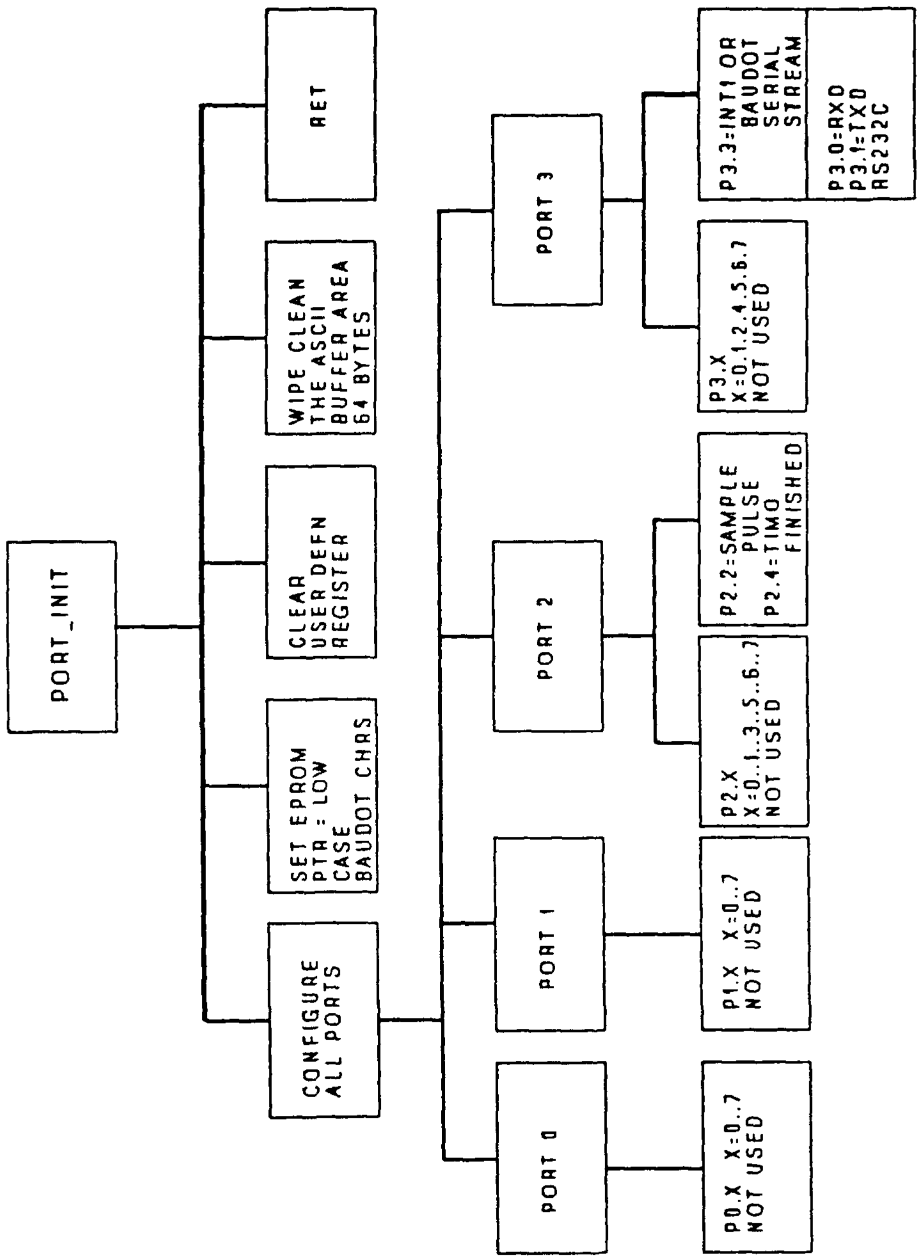


FIG 4.2.2 ROUTINE TO INITIALISE THE MICROCONTROLLERS PORTS

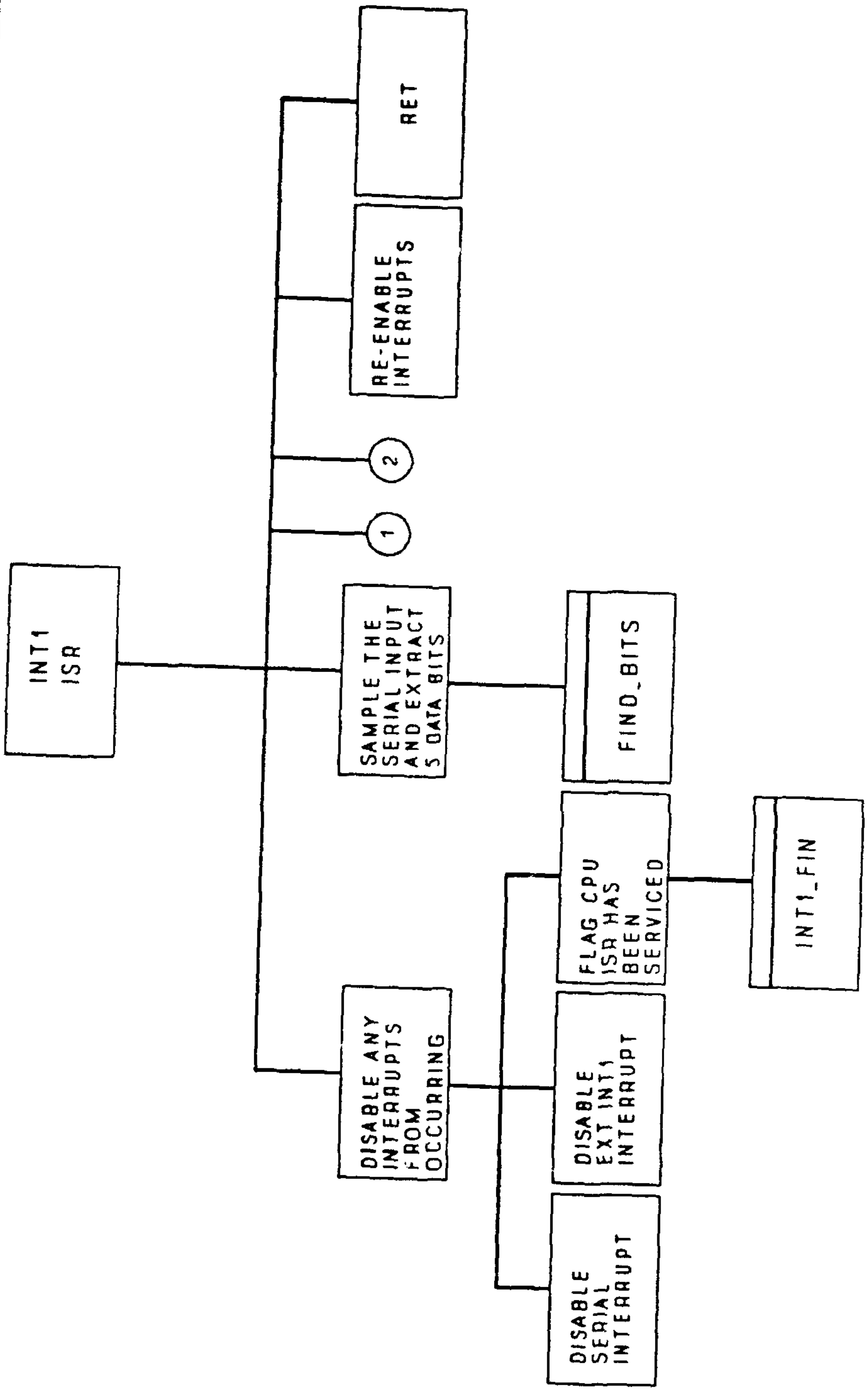


FIG 4.2.3a EXTERNAL INTERRUPT 1 SERVICE ROUTINE

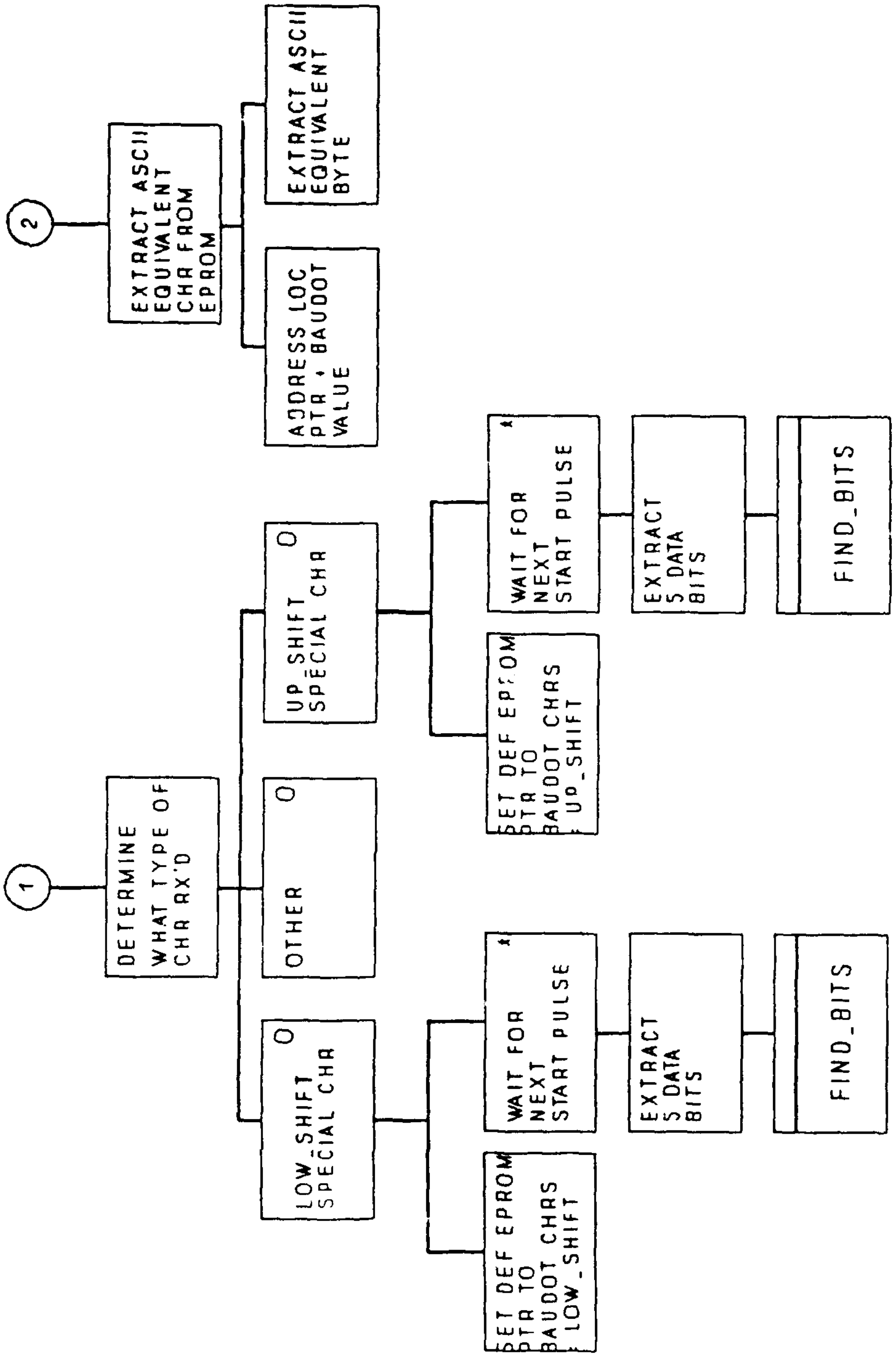


FIG 4.2.3b EXTERNAL INTERRUPT 1 SERVICE ROUTINE

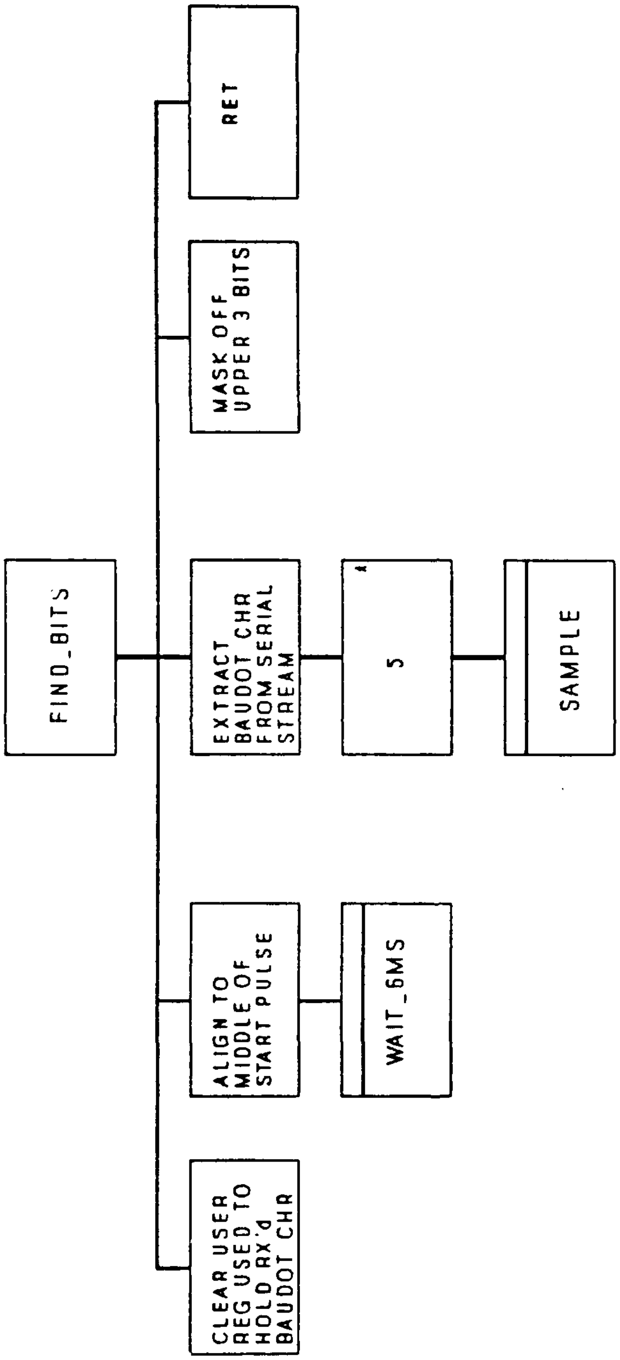


FIG 4.2.4 ROUTINE TO READ A BAUDOT CHARACTER

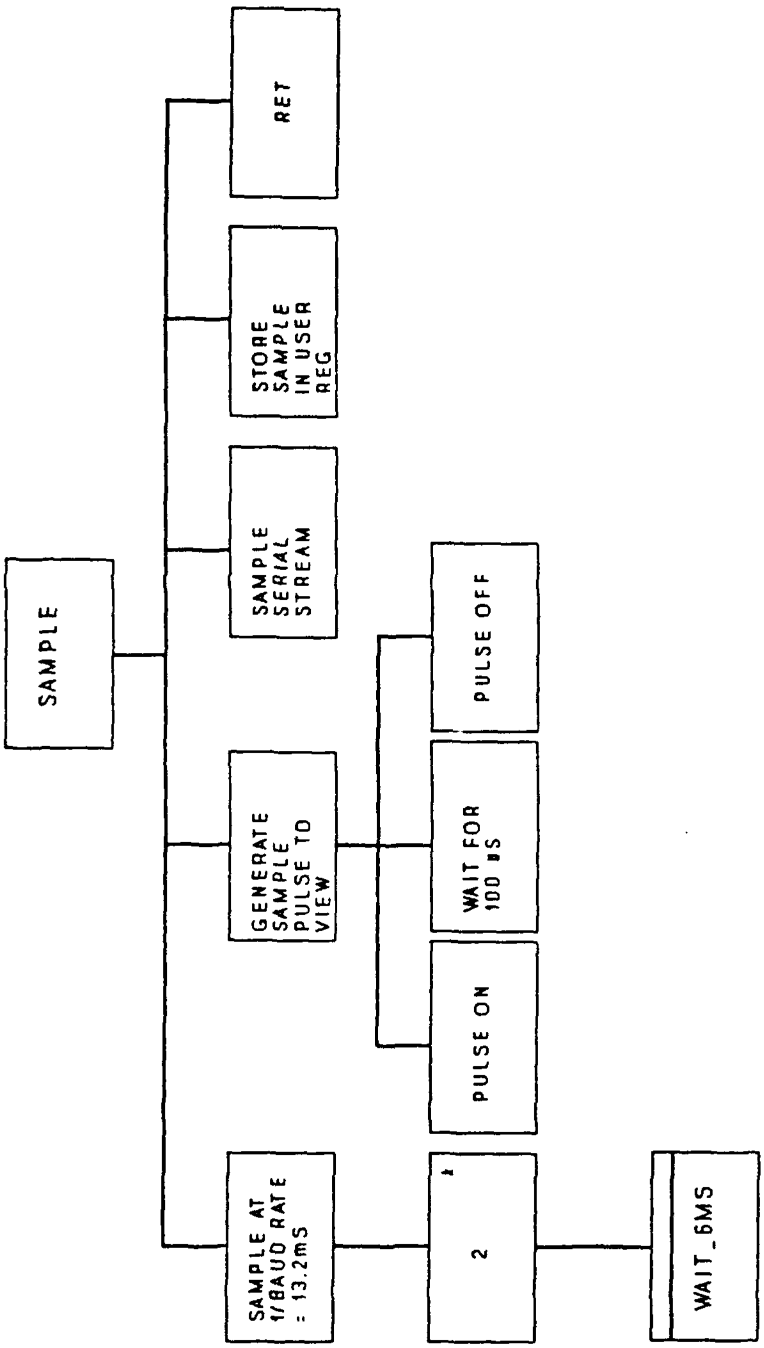


FIG 4.2.5 ROUTINE TO SAMPLE THE BAUDOT RS232 SERIAL STREAM

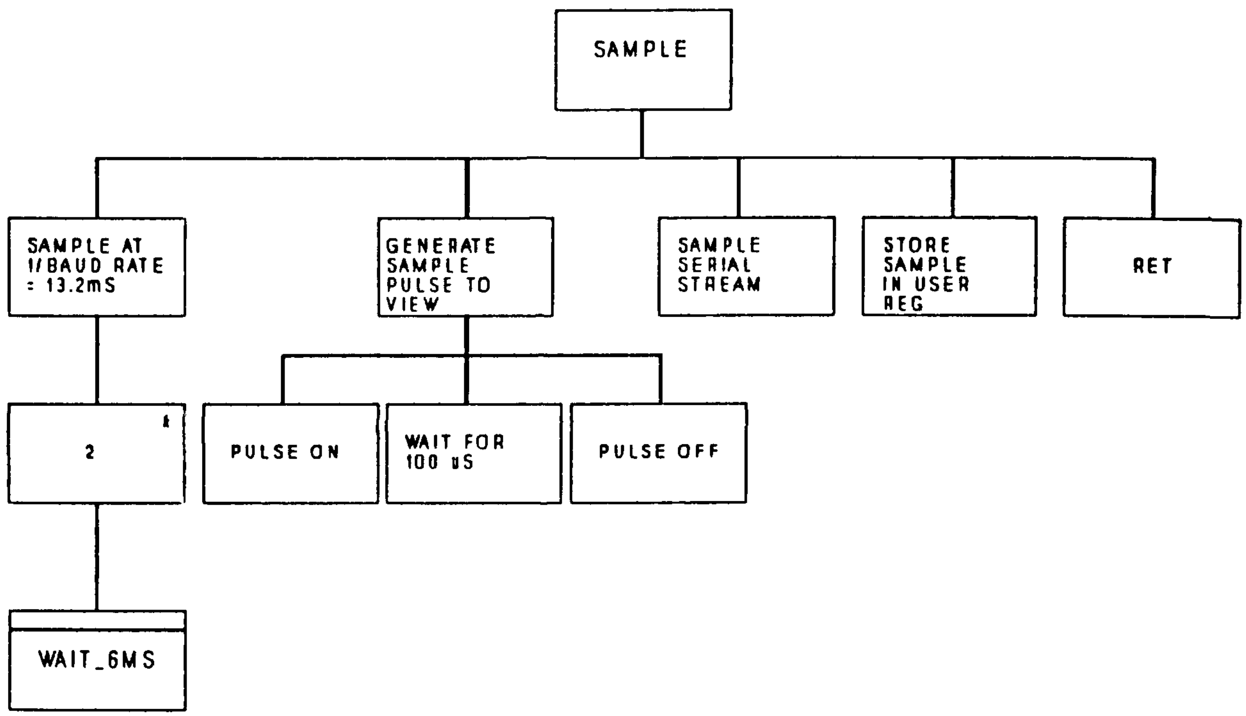


FIG 4.2.5 ROUTINE TO SAMPLE THE BAUDOT RS232 SERIAL STREAM

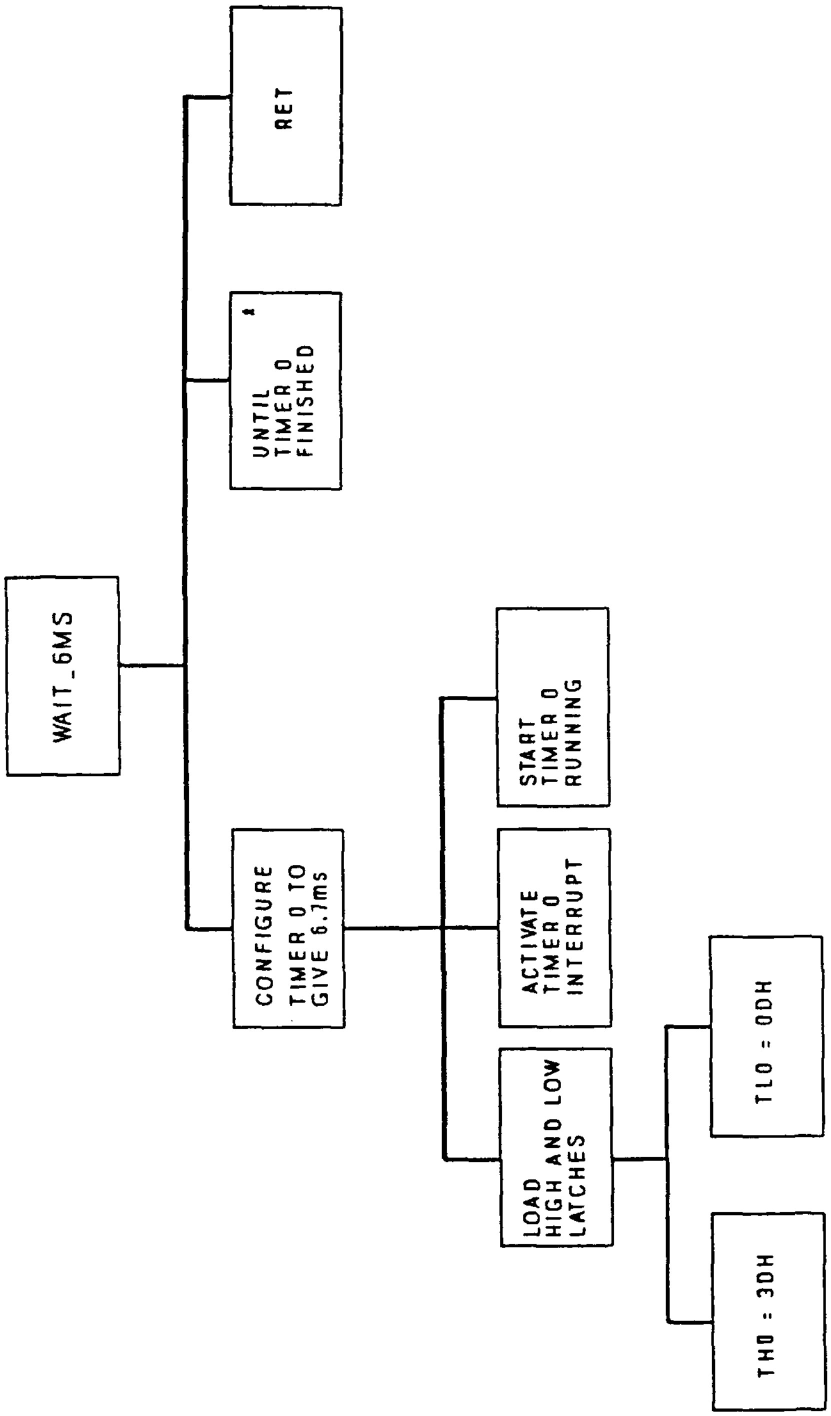


FIG 4.2.6 ROUTINE TO GENERATE A 6.65mS TIMING INTERVAL

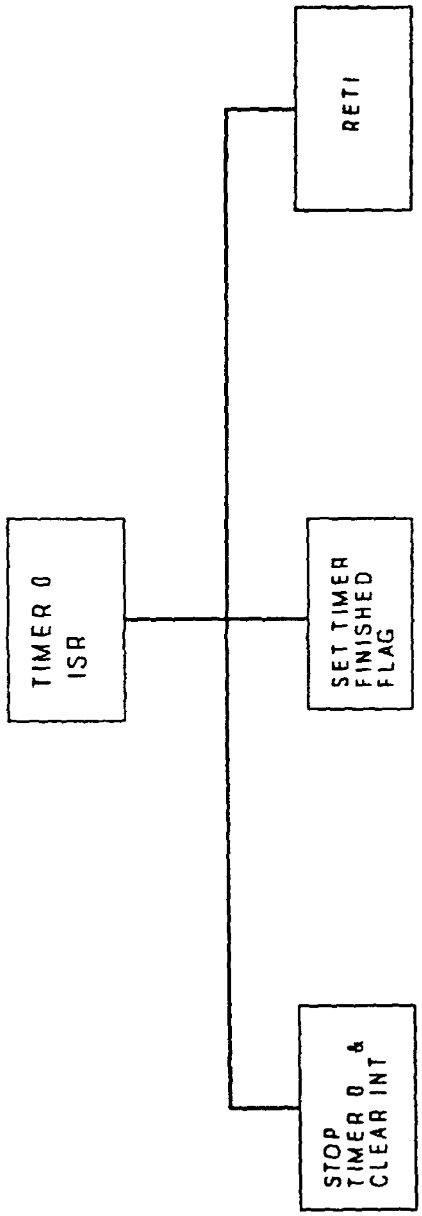


FIG 4.2.7 TIMER 0 INTERRUPT SERVICE ROUTINE

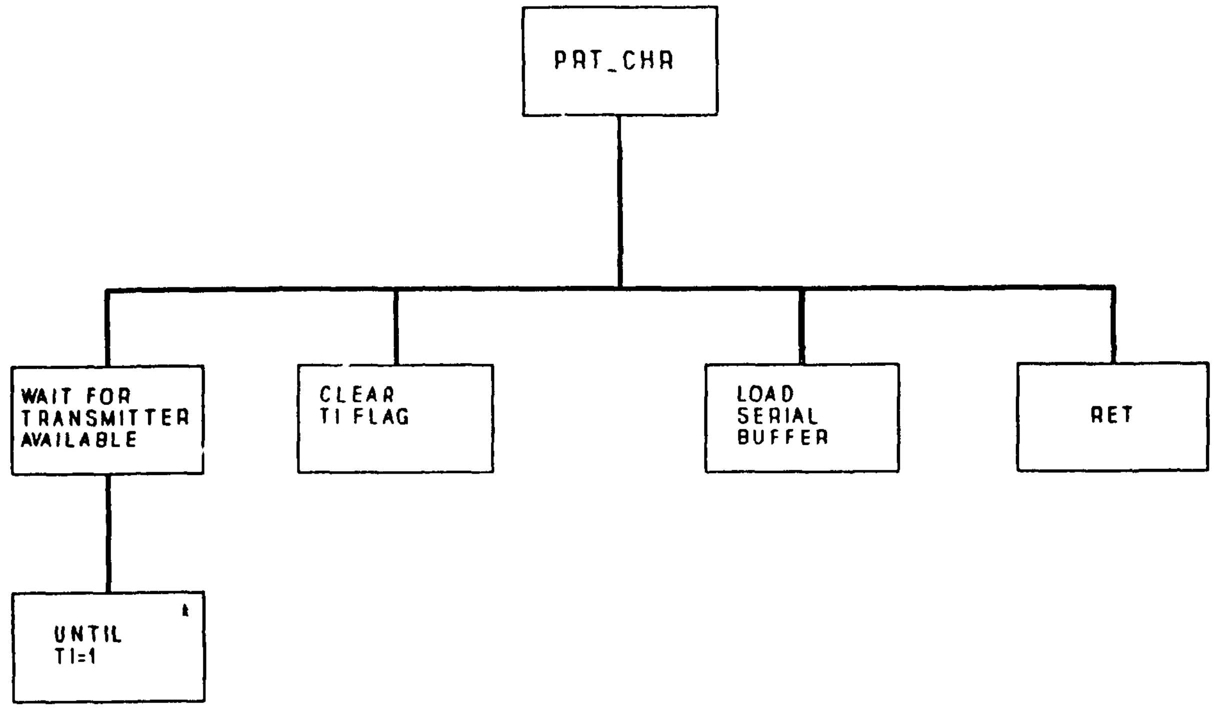


FIG. 4.2.8 ROUTINE TO OUTPUT AN ASCII CHARACTER

APPENDIX A

CCITT alphabet number 2 (the Baudot code)

BIT 1	BIT 2	BIT 3	BIT 4	BIT 5	LOWER CASE	UPPER CASE
0	0	0	0	0	no action	no action
1	0	0	0	0	T	5
0	1	0	0	0	CR	CR
1	1	0	0	0	O	9
0	0	1	0	0	SPACE	SPACE
1	0	1	0	0	H	
0	1	1	0	0	N	,
1	1	1	0	0	M	.
0	0	0	1	0	LF	LF
1	0	0	1	0	L	)
0	1	0	1	0	R	4
1	1	0	1	0	G	
0	0	1	1	0	I	8
1	0	1	1	0	P	0
0	1	1	1	0	C	:
1	1	1	1	0	V	=

CCITT alphabet number 2 (the Baudot code)

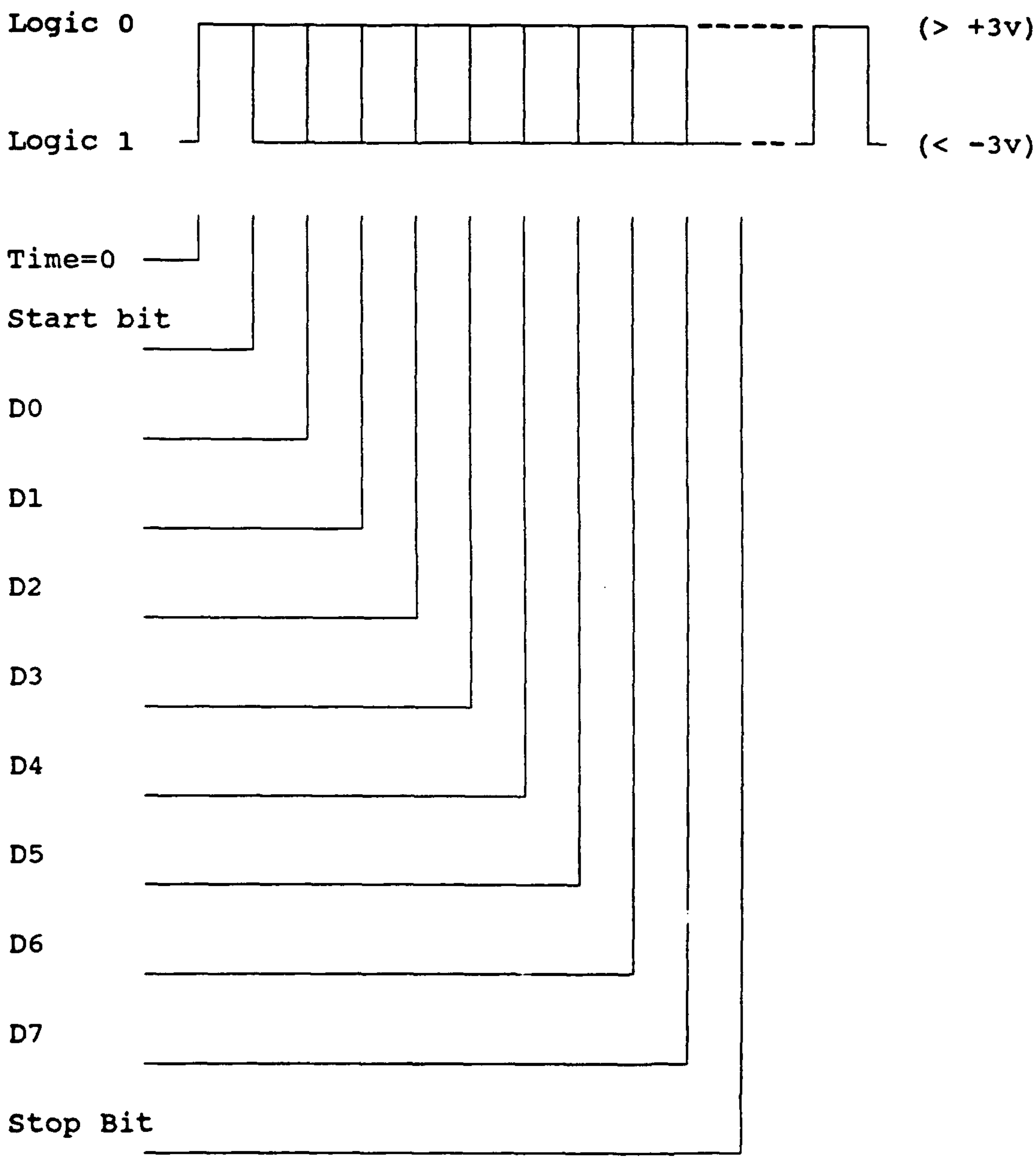
BIT 1	BIT 2	BIT 3	BIT 4	BIT 5	LOWER CASE	UPPER CASE
0	0	0	0	1	E	3
1	0	0	0	1	Z	+
0	1	0	0	1	D	who are you
1	1	0	0	1	B	?
0	0	1	0	1	S	'
1	0	1	0	1	Y	6
0	1	1	0	1	F	
1	1	1	0	1	X	\
0	0	0	1	1	A	-
1	0	0	1	1	W	2
0	1	0	1	1	J	BELL
1	1	0	1	1	UPPER SHIFT	UPPER SHIFT
0	0	1	1	1	U	7
1	0	1	1	1	Q	1
0	1	1	1	1	K	(
1	1	1	1	1	lower shift	lower shift

Note: Unallocated combinations are used for specific national symbols and should, therefore, never be used for international transmission

APPENDIX B

RS-232C serial data-byte timing waveform

Format	:	Baud Rate :	9600 (Pulse width =104uS)
		Data Bits :	8
		Start Bits:	1
		Stop Bits :	1

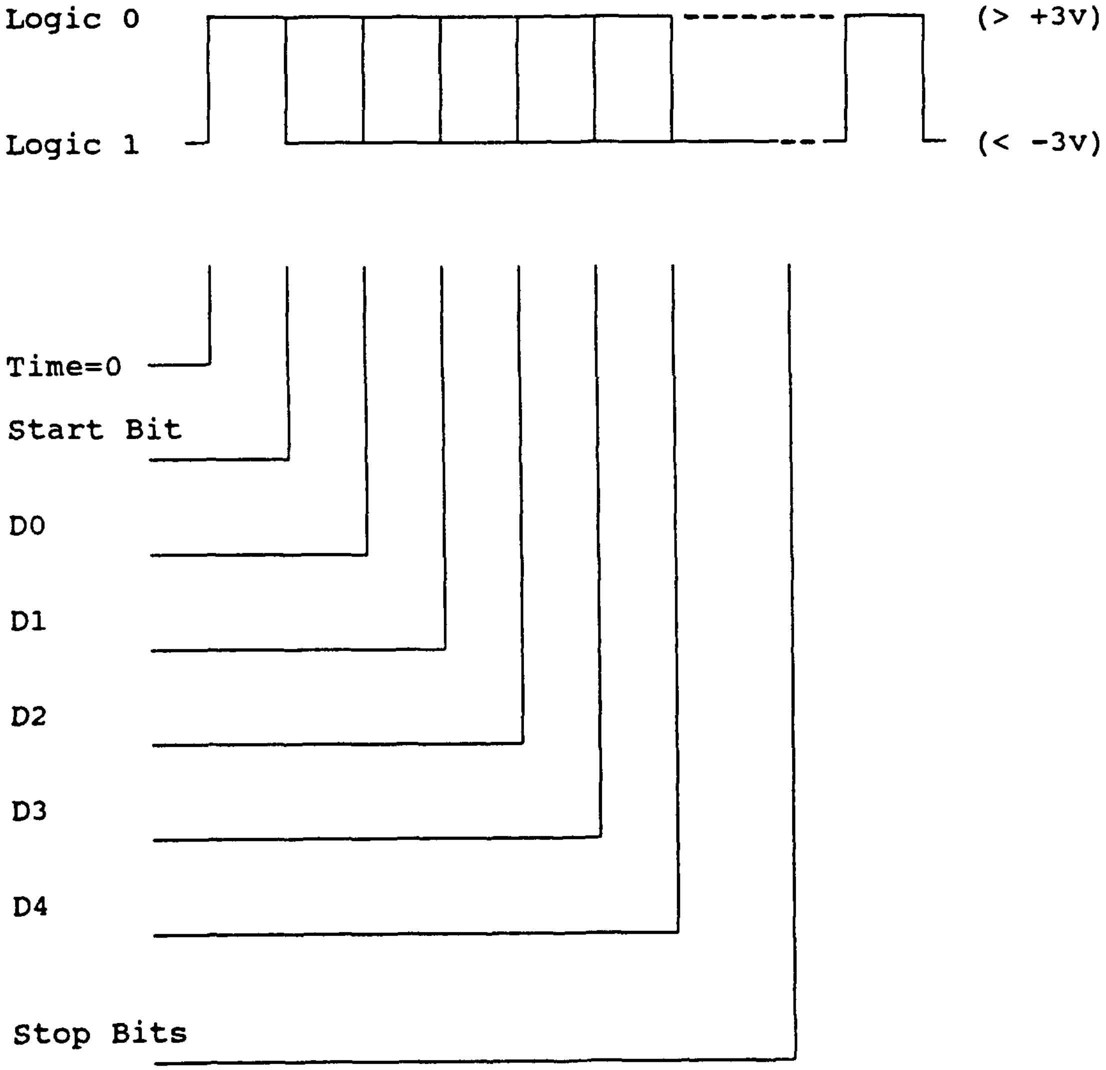


INTENTIONALLY BLANK

APPENDIX C

CCITT alphabet number 2 (Baudot code) data_byte timing waveform

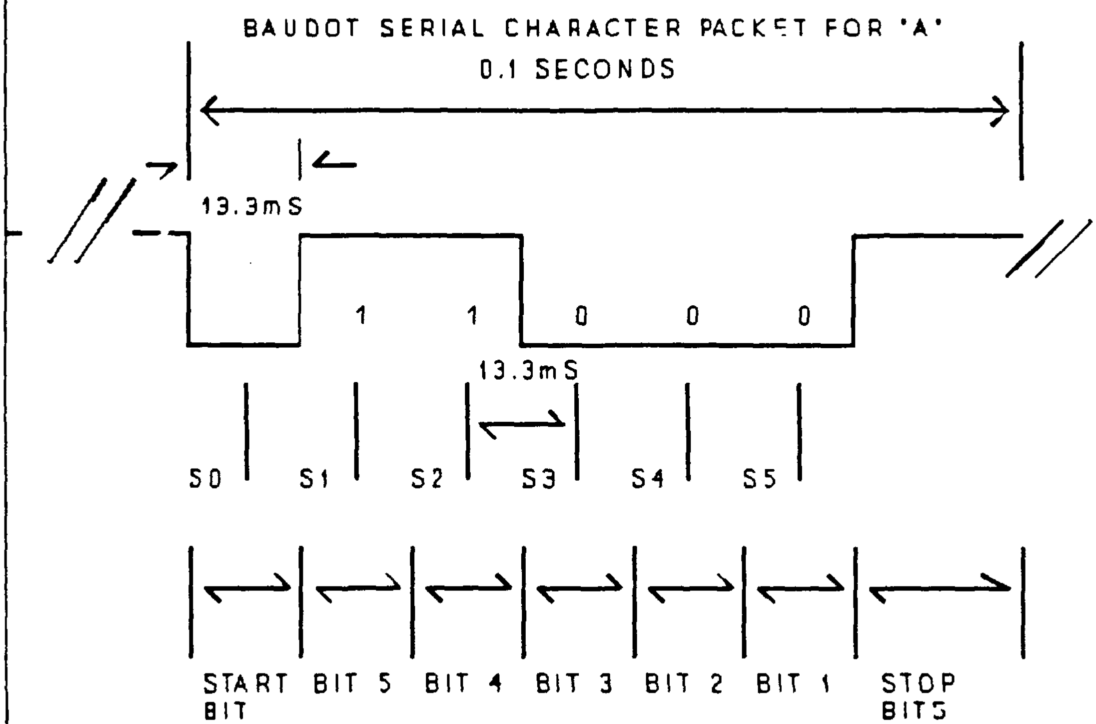
Format :	Baud Rate :	75	(Pulse width = 13.3mS)
	Data Bits :	5	
	Start Bits:	1	
	Stop Bits :	1.5	



INTENTIONALLY BLANK

APPENDIX D

EXTRACTION OF BAUDOT CHARACTER



INTENTIONALLY BLANK

APPENDIX E

OPERATING INSTRUCTIONS

[1]. Connect the +240V supply to the 3-pin plug on the rear of the box housing, using the white mains cable and switch ON.

[2]. (a). Connect the shorter of RS232 cables (Female D-Type connector on both ends), to the TOP Male D-Type connector (labelled ASCII) on the front of the Box Housing.

(b). Connect the other end of this shorter cable to the device which is generating the ASCII Characters, for example the ELF.

[3]. (a). Connect the longer of RS232 cables (Female D-Type connector on one end and Male D-Type on the other) to the BOTTOM Male D-Type connector (labelled BAUDO[T]) on the front of the Box Housing.

(b). Connect the other end of this cable to the device which is to
receive and transmit the BAUDOT Characters.

[4]. (a). The ASCII Character generating Terminal must be configured to
(1). BAUD RATE of 9600 Baud.
(2). 8 Data Bits.
(3). One Stop Bit and One Start Bit.

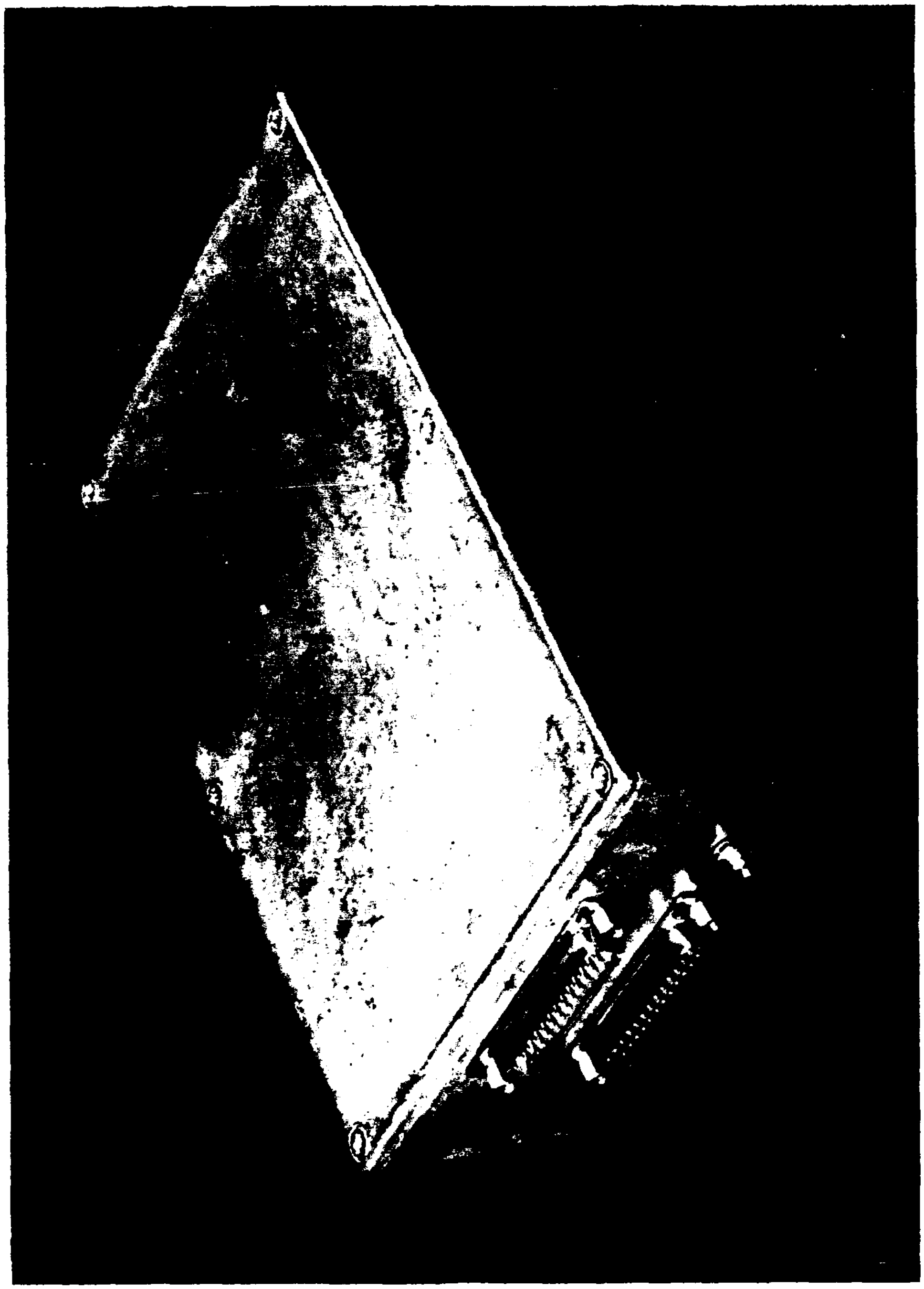
(b). The BAUDOT Character generating Terminal must be configured to
(1). BAUD RATE of 75 Baud.
(2). 5 Data Bits
(3). One Start Bit and One and a Half Stop Bits.

[5]. When the cable connections have been made Press the RESET Button on the front of the Box Housing. The Unit is now operating.

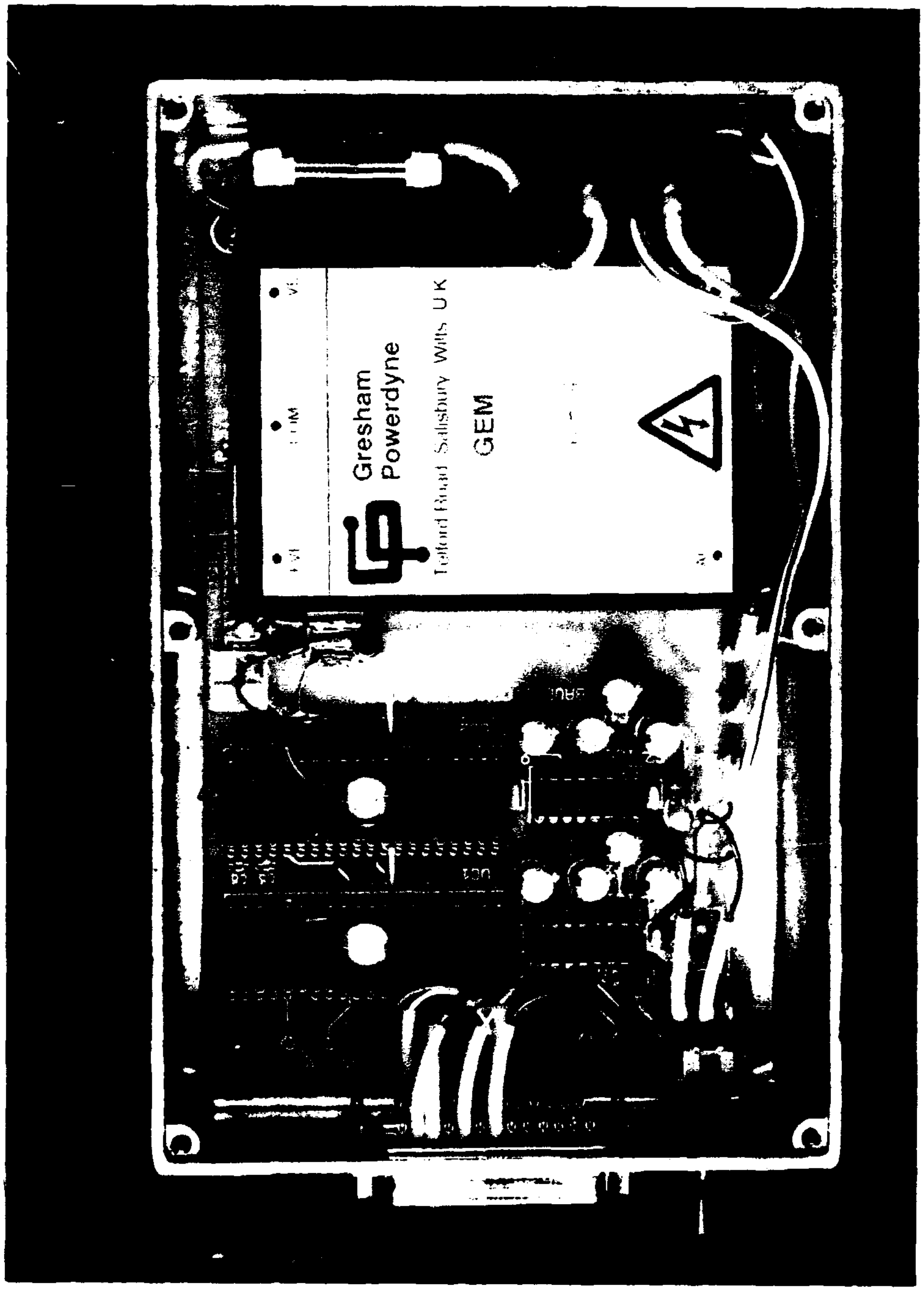
INTENTIONALLY BLANK

APPENDIX F

PHOTOGRAPH SHOWING THE EXTERIOR OF THE APPLIQUE UNIT



PHOTOGRAPH SHOWING THE INTERIOR OF THE APPLIQUE UNIT



REPORT DOCUMENTATION PAGE

DRIC Reference Number (if known) ....

Overall security classification of sheet ....UNCLASSIFIED..

(As far as possible this sheet should contain only unclassified information. If it is necessary to enter classified information, the field concerned must be marked to indicate the classification eg (R), (C) or (S).

Originators Reference/Report No.	Month	Year
MEMO 4479	MAY	1991

Originators Name and Location
RSRE, St Andrews Road
Malvern, Worcs WR14 3PS

Monitoring Agency Name and Location

Title
A BI-DIRECTIONAL BAUDOT/ASCII CODE CONVERTER

Report Security Classification	Title Classification (U, R, C or S)
UNCLASSIFIED	U

Foreign Language Title (in the case of translations)

Conference Details

Agency Reference	Contract Number and Period

Project Number	Other References

Authors	Pagination and Ref
ANDERSON, L	vp

Abstract

A unit has been developed to interface between a terminal generating BAUDOT Code at 75 Baud and a terminal generating ASCII Code at 9600 Baud, facilitating bi-directional serial communication. This report describes the hardware and software design; and includes the Operating Instructions.

Abstract Classification (U,R,C or S)
U

Descriptors

Distribution Statement (Enter any limitations on the distribution of the document)

UNLIMITED

580/48

INTENTIONALLY BLANK